UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR
GRADO EN INGENIERÍA INFORMÁTICA

ANÁLISIS DE DATOS SOBRE LA PLATAFORMA MAGENTO

Autor: Daniel Lozano Cofrades
Tutor: Fernando Fernández Rebollo
07/10/2014

Agradecimientos

En primer lugar agradecer todo el apoyo que me han prestado mis padres, Fco. Javier y Susana, durante todos estos años de carrera. De no ser por ellos, no hubiese sido posible llegar hasta aquí, muchas gracias.

También dar las gracias a Laura por todo el apoyo prestado, sobre todo por aguantarme en los momentos duros de este camino. Gracias por brindarme la posibilidad de exiliarme contigo concluir esta memoria. Tusen takk.

Por último, a mi tutor Fernando Fernández Rebollo por guiarme durante la realización de este TFG incluso estando de vacaciones.

Resumen

El mundo del comercio electrónico y en consecuencia, las tiendas on-line lleva experimentando a lo largo de estos últimos años un enorme crecimiento con respecto a las tiendas físicas. Esto es debido a que el usuario tiene la posibilidad de comprar productos desde cualquier dispositivo inteligente como ordenadores portátiles, Tablets o Smartphones, que constituyen elementos esenciales en el desarrollo diario de sus actividades. Esto ha dado lugar a plataformas de comercio electrónico como Magento que permiten crear y llevar el control de forma sencilla sobre tu tienda on-line.

Por otra parte, estamos en la era dorada del análisis de datos. El análisis de datos consiste básicamente en examinar los datos en bruto con el fin de sacar conclusiones y se utiliza hoy en día para tomar mejores decisiones empresariales. Algunas de estas herramientas son libres, como Weka y permiten la ejecución de estas técnicas de manera sencilla. Debido a la informatización masiva que ha habido en los últimos años, las empresas cuentan con un inmenso volumen de datos lo que permite que este tipo de técnicas tengan todavía mayor repercusión.

De la unión de estos puntos nace este Trabajo de Fin de Grado, con el objetivo de crear una prueba de concepto con el propósito de verificar la posibilidad de integrar una plataforma de aprendizaje automático sobre la plataforma de gestión de tiendas y comercio electrónico Magento. De esta forma, se estudian las diferentes alternativas de diseño así como las ventajas y limitaciones de utilizar la tecnología elegida.

Abstract

The world of e-commerce and therefore, on-line stores, have experienced tremendous growth over the years compared to physical stores. This is because the user has the possibility of purchasing products from any smart device such as laptops, tablets or smartphones; which constitute essential elements in the daily development of its activities.

On the other hand, we are in the golden age of the data analysis. The analysis of data basically consists in examining the raw data in order to draw conclusions. It's used today to make better business decisions.
Due to massive computing that has occurred in recent years, the companies have a huge volume of data which allows this type of techniques have still greater impact.

This Bachelor Thesis is born for joining both thematic areas, with the aim of creating a proof of concept in order to verify the possibility of integrating a machine learning tool on Magento, an e-commerce management platform of stores. In this way, the different alternatives of design as well as the advantages and limitations of using the technology chosen will be studied.

Índice de contenidos

Índice de figuras

Índice de tablas

Glosario de términos

On-line: relativo a que está disponible o se realiza a través de internet

Smartphone: (teléfono inteligente) es un teléfono móvil construido sobre una plataforma informática móvil, con una mayor capacidad de almacenar datos y realizar actividades semejantes a una minicomputadora.

Tablet: computadora portátil integrada en una pantalla táctil de mayor tamaño que un Smartphone.

Discretización: en el contexto del análisis de datos, hace referencia a una técnica de preprocesado en la que se equipara el rango o dominio de un dato, al de otro, con el fin de equiparar la repercusión de ambos datos en el análisis.

Cluster: (inglés) utilizado en inteligencia artificial para definir cada uno de los grupos cuando se aplica una técnica de agrupamiento o *clustering*.

Magento: gestor de contenidos web de código abierto para comercio electrónico.

Weka: colección de algoritmos de aprendizaje automático.

PYMES: abreviatura de *Pequeñas Y Medianas Empresas.*

Sistema Esclable: es un sistema que tiene capacidad para reaccionar y adaptarse sin perder calidad en los servicios ofrecidos.

Sistema Flexible: es un sistema que tiene capacidad para adaptar sus características a las necesidades del trabajo a realizar.

Licencia GNU: licencia que garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir (copiar) y modificar el software.

Licencia AGPL: licencia copyleft derivada de la Licencia Pública General de GNU diseñada específicamente para asegurar que en caso de que el software corra en servidores de red, será necesaria la cooperación con la comunidad.

Tupla: secuencia ordenada de objetos que representa información, en este caso representan una instancia del conjunto de datos.

Active Record: patrón de arquitectura de comunicación de algunos lenguajes de comunicación con la Base de Datos.

Minería de datos: es un campo de las ciencias de la computación referido al proceso que intenta descubrir patrones en grandes volúmenes de conjuntos de datos.

Inteligencia de negocio: conjunto de estrategias y aspectos relevantes enfocadas a la administración y creación de conocimiento sobre el medio, a través del análisis de los datos existentes en una organización o empresa.

Inteligencia artificial: área multidisciplinaria que, a través de ciencias como la informática, la lógica y la filosofía, estudia la creación y diseño de entidades capaces de razonar por sí mismas utilizando como paradigma la inteligencia humana.

BlazeDS: tecnología de mensajería asincrónica para componentes de servidor desarrollados en Java que permite a los desarrolladores conectar aplicaciones Adobe Flex y Adobe AIR con entornos empresariales Java.

Java: lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

Mapear: convertir o transformar una serie de datos en un formato determinado.

API: (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.

Datas set: (inglés). Conjunto de datos.

Copyleft: una práctica que consiste en el ejercicio del derecho de autor con el objetivo de permitir la libre distribución de copias y versiones modificadas de una obra u otro trabajo, exigiendo que los mismos derechos sean preservados en las versiones modificadas.

1 Introduction

This work is contained in the context of e-commerce and business intelligence. Magento is a web content platform for e-commerce that allows to create online stores, as well as manage them. The platform has been created to generate basic tasks like add and manage customers and products, but also has features more complex as displaying statistical data concerning sales and products as you can be seen in Figure 1 where the Magento admin panel is displayed:

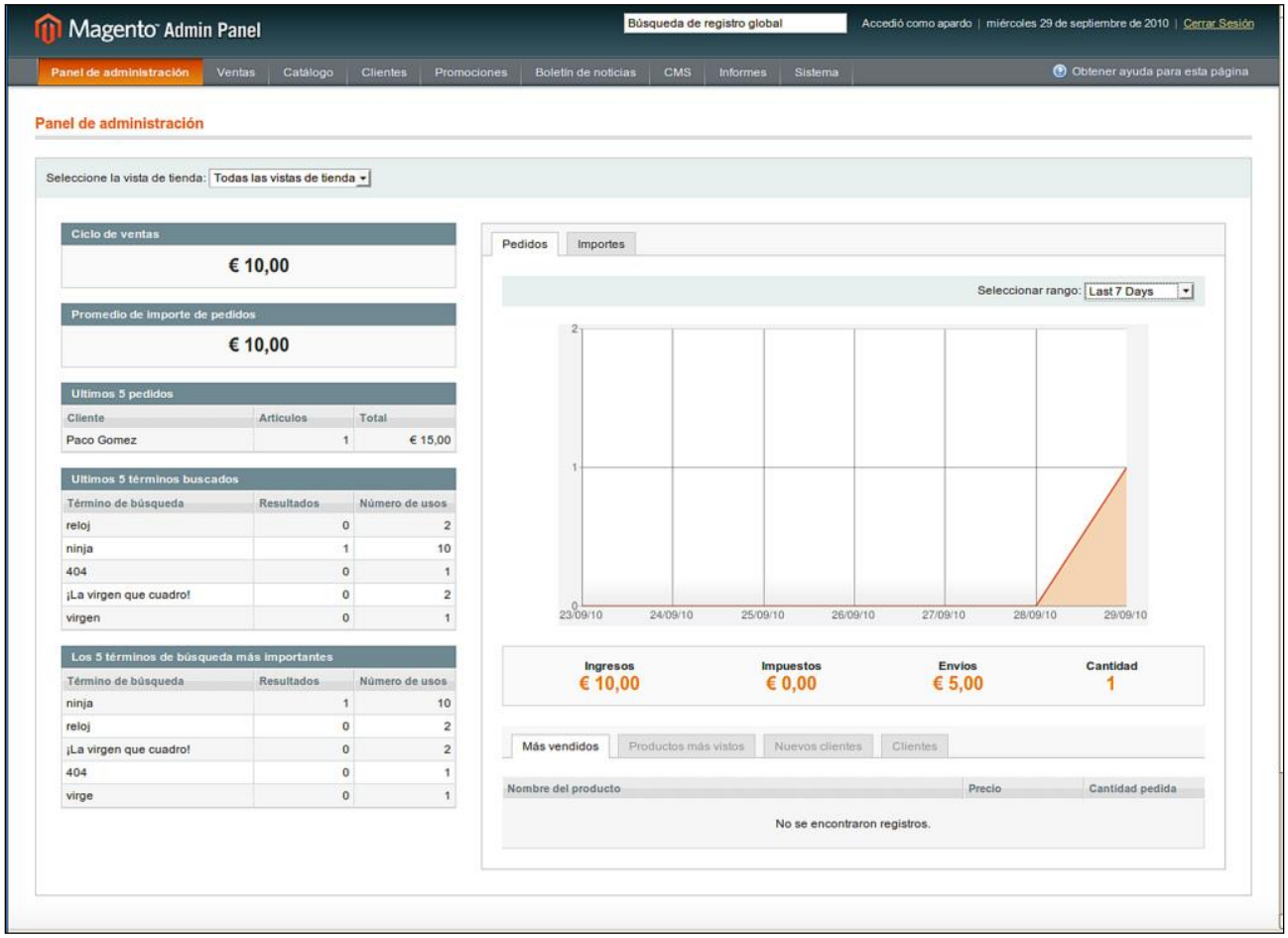


Figura 1: Panel de Administración de Magento

This admin panel contains information regarding more search terms, total sales, etc. And this information is very useful because the store manager can use these data to, for example, add a product to the inventory because users have searched it repeatedly.

1.1 Problem definition

A main part of the Magento platform are the stats options that are displayed in the main panel about the elements of the trading platform. This information refers mainly to the 3 pillars of e-commerce platforms or e-commerce platforms: products, customers and sales. These sections reflect relationships between them as best sellers or customer with greater volume of expenditure on purchases, etc. Also shows the most searched words or latest products sold, which can serve to the Manager of the store as a reference to make simple decisions. But, in the commercial sector, doesn't report a very large volume of data is usually managed and these small grants not reporting enough useful information, and most importantly, personalized.
Analysing the platform, it has detected that it would be helpful to make data analysis with some level of customization to obtain useful information for different cases involving decision-making support by the Manager of the store.

In short, there are 3 main problems:

1. The poverty of the reported statistics by Magento (only cumulative statistics).

2. Inability to customize the reported statistics.

3. Inability to generate complex analytical reports as support for decision-making, which are very useful for this kind of platforms.

1.2 Objectives

The main purpose of this work is the development a client/server platform as a proof of concept to perform predictive analysis on the data of Magento platform.

1. Enable to perform a data pre-processing in order to customize the obtained reports.

2. Include predictive analysis methods for complex reports useful to managers of the platform.

This objectives, are linked to other partial sub-objectives:

- Acquire knowledge about Magento platform.

- Deepen knowledge about Weka and Machine learning tools.

- Study and analyse Magento and Weka APIs.

- Integrate the developed system.
- Evaluate de developed system.

1.3 Socio-economic Environmenty and Regulatory Framework

Data analysis is a discipline that is currently expanding due to the large volume of analysable data that are collected on a daily basis. Increasingly, companies choose this technique when it comes to business decision-making support. Currently, this business model is an unavoidable option for companies because, roughly, these techniques consist in "monetize" the stored data, which leads to higher profits in business. As a Manager of Information Technologies to the big Spanish company Acciona Energia said:

"Data has always been, but now technology allows a higher level of storage and analysis"

Javier Arratibel, Manager of Information Technologies to Acciona Energía.

The regulatory framework would be mainly supported by the Organic Law 15/1999 of 13 December on the Protection of Personal Data (LOPD). The object of this law is to guarantee and protect, with regard to the processing of personal information or public liberties and fundamental rights of physical persons, especially in order to preserve the honor, personal and family privacy and the full exercise of personal rights against alteration, loss, treatment or unauthorized access.

All this applies to personal data recorded on any type of hardware candidate for treatment (whether computerized or manual). Therefore, on the occasion of the entry into force of LOPD raises a number of obligations for those public or private entities that have files with personal information.

All the software used for this project was under GNU General Public License. This is a copyleft license and guarantee the freedom to use, study, share (copy) and modify the software.

1.4 Development phases

The wholeness of the project is included in a serie of specific development stages shown below:

State of Art

At this stage, the framework in which the problem is contextualized is studied. Different platforms and data analysis technologies will be explored to implement the system in order to select the most appropriate technology.

Problem definition

This phase studies the context in which the problem is framed. The technologies and shortcomings of Magento are examined in depth in order to select the most suitable to develop this environment.

Solution

At this stage the solution is developed, the essential features and the detailed features are identified.

Evaluation

In this step, an assessment is carried out to validate the implemented solution, showing that the solution meets the requirements defined in the previous section.

1.5 Resources employed

The resources used to develop the solution obtained can be divided into two main categories: literature sources for understanding and use both e-commerce platforms and different data analysis techniques; and development tools and documentation needed to develop the solution.
The literature sources used were are follows:

- Google Scholar.
- University Carlos III of Madrid's library. Escuela Politécnica Superior.

The development tools and documentation used were:

- Tools:
  - Ruby 1.9.3 p484 (2013-11-22)
  - Rails 4.0.2
  - Magento 1.8.1.0
  - Weka 3.6.11

- Documentation:
  - Magento API
  - Weka API
  - API RubyOnRails

1.6 Document structure

This document has been written using the following chapters:

- **First chapter:** aims to introduce the work done and the document content.
- **Second chapter:** the state of the art, establishes the context in which the problem is framed in this final thesis. In this section, reviewing the predictive analysis methods existing nowadays and the technologies that will develop the solution
- **Third chapter:** software project management, reflects the project scope, work plan, resource management, risk management and test plan.
- **Forth chapter:** solution, details the solution conducted, describing the adopted type and the development process.
- **Fifth chapter:** evaluation, demonstrates the validity of the developed solution, checking through tests and/or questionnaires, evaluating whether the problems are resolved and whether the objectives exposed in the introduction are satisfied.
- **Sixth chapter:** conclusion, provide summary of work and personal view of it.
- **Seventh chapter:** bibliography, lists references used in the performance of the work.
- **Eighth chapter:** refers to several annexes and version control, contains more detailed information about several points.

2 El estado de la cuestión

El objetivo de este apartado es el de establecer el contexto en el que se enmarca el problema que se aborda en el presente trabajo fin de grado. En primer lugar, se estudiará el marco tecnológico así como las diferentes herramientas de análisis de datos presentes hoy en día. En segundo lugar, se determinará que tecnología se adapta mejor para desarrollar la solución.

2.1 Análisis Predictivo como parte de la Inteligencia de negocio

El Análisis Predictivo [APR] es una técnica que utiliza algoritmos de minería de datos o *data mining* junto con estadística. Se basan en el análisis de los datos actuales e históricos para hacer predicciones sobre futuros eventos.

> *El Data Mining es un proceso que permite extraer conclusiones confiables sobre las condiciones presentes y los eventos futuros, a través de la aplicación de métodos estadísticos, matemáticos y de reconocimiento de patrones.*
>
> *Gartner Group*

Esta práctica se lleva a cabo en la inteligencia de negocio o *business intelligence* [BIN] como una estrategia enfocada a la creación de conocimiento sobre el medio de actuación a través de los datos de la propia empresa. Este tipo de estrategias son una parte fundamental como soporte para toma de decisiones y ofrecen un respaldo basado en históricos, con el fin de que dichas decisiones sean lo más eficaces posible.

Existen 3 modelos principales de análisis predictivo:

- **Modelos predictivos:** Esta clase de modelos analizan los resultados anteriores para evaluar posibles comportamientos futuros. En una plataforma de comercio electrónico como Magento, serviría para predecir qué productos pueden ser los más vendidos o que ventas se van a obtener en una época concreta basado en la evolución anterior.

- **Modelos descriptivos:** Esta clase de modelos describen relaciones en los datos para poder clasificar (con unas clases ya definidas) o agrupar (definiendo un número de grupos) datos en función de sus características. Este modelo podría ser muy útil a la hora de clasificar los productos por una serie de familias predefinidas o agrupar los clientes de características similares con el fin de generar ofertas más personalizadas.

- **Modelos de decisión:** Esta clase de modelos describen la relación entre todos los elementos de una decisión. Los datos conocidos (incluidos los resultados de los modelos de predicción), la decisión y el plan de variables y valores que determinan la decisión – con el fin de predecir los resultados de las decisiones de muchas variables. Estos modelos pueden ser utilizados en optimización. Aplicado a la plataforma de comercio electrónico Magento, se podrían determinar los "productos relacionados" existentes en las tiendas on-line.

El ciclo del análisis predictivo se presenta de la siguiente forma:

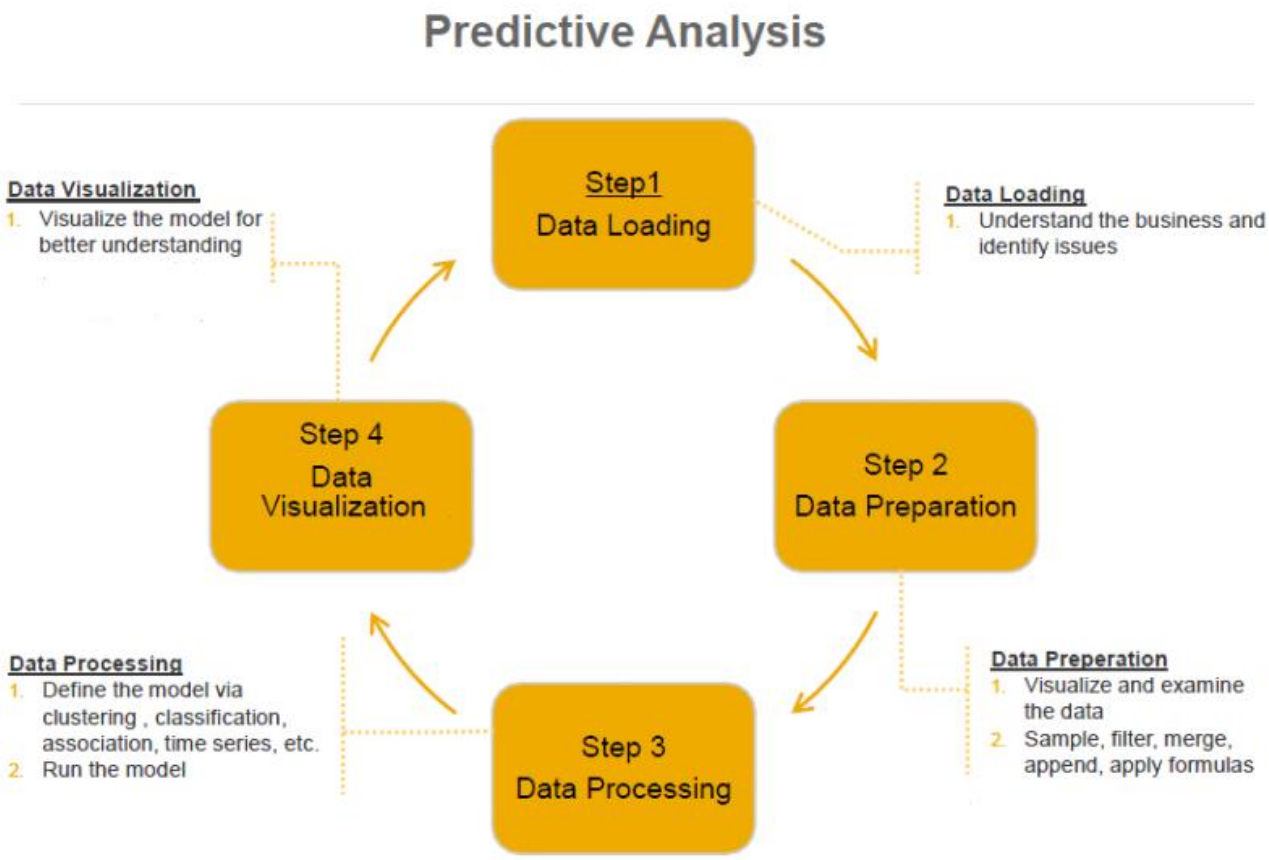


Figura 2: Ciclo del Análisis Predictivo

Como se puede observar en la figura, el análisis predictivo es un proceso iterativo que cuenta con un ciclo perfectamente definido. En primer lugar, como se puede apreciar *(Step 1)*, se selecciona un conjunto de datos a tratar del que queramos sacar alguna conclusión. El segundo paso *(Step 2)*, es la preparación de los datos; en este caso se realiza la parte del preprocesado de los datos, con el fin de prepararlos para el procesamiento de los mismos. Estas tareas pueden ser, por ejemplo, la exclusión de valores pertenecientes al conjunto para descartar el posible ruido producido por atributos no relevantes o descartables por motivos concretos; o la discretización de valores para

determinarles el mismo peso a otros atributos del conjunto.
El tercer paso *(Step 3)* constituye la parte de la generación del modelo, así como su aplicación sobre los datos. El cuarto paso (*Step 4*), es la visualización o análisis del modelo y las consecuencias de su aplicación sobre los datos. Este tipo de prácticas necesitan una interpretación posterior al análisis y normalmente una decisión viene determinada por diferentes pruebas y comparación con los modelos anteriores. Por esa razón, en la Figura 2 se expresa el proceso del análisis de datos con un diagrama cíclico.

Una parte fundamental del proceso es la creación del modelo. El modelo será la pauta del procesado de los datos, es decir, el modelo define qué datos del conjunto preprocesado pertenecen a qué clase en una tarea de clasificación, o qué dato entra dentro de cada *cluster* (o grupo) en una tarea de agrupamiento. Por lo que la generación del modelo no es algo trivial y es necesario tener experiencia y hacer una buena interpretación de los datos en el preprocesado para generar modelos útiles aplicables.
Para la **generación de los modelos** se utilizan una serie de **algoritmos de Inteligencia Artificial** aplicados sobre los conjuntos de datos de los que se busca sacar conclusiones. Existen numerosos algoritmos para la generación de estos modelos y se encuentran agrupados en numerosas plataformas. Existen numerosas librerías y programas con los que crear los modelos. Todas estas plataformas se analizarán en el apartado 2.4 Elección de la tecnología de desarrollo de la aplicación cliente-servidor.

2.2 Aprendizaje automático como disciplina para generación de modelos

El aprendizaje automático o *machine Learning* [MLR], es una rama dentro de la Inteligencia Artificial dedicada al estudio de agentes/sistemas capaces de aprender o evolucionar en base a unas evidencias dadas. El proceso principal consiste en partir de una evidencia conocida, crear una hipótesis y dar respuesta a situaciones nuevas no conocidas. Estas hipótesis se generan por medio de una serie de algoritmos que a continuación clasificaremos en función del tipo de salida de los mismos. **El marco que engloba este proyecto, se centra en la generación de modelos por medio de 2 tipos de aprendizaje:**

- **Aprendizaje Supervisado:** al algoritmo produce una función determinada por la correspondencia entre las entradas y las salidas deseadas. Se llama supervisado porque se determina la salida deseada dada una entrada, por tanto hay una supervisión externa que determina qué entradas pertenecen a qué clases. Como se determina la salida, un ejemplo muy claro de este tipo de tareas es la **clasificación**, dado que en función de una entrada, se determina la clase de pertenencia en función de las evidencias anteriores.

- **Aprendizaje No Supervisado:** se modela únicamente en función de las entradas del sistema y se generan patrones y relaciones entre los datos para determinar los clusters o grupos. Como no se determina la salida, un ejemplo muy claro de este algoritmo es el *clustering* o **agrupamiento**, dado que se generan los grupos en función de las relaciones entre entradas.

Por medio de estas dos técnicas se puede realizar Análisis Predictivo de tipo descriptivo. Este tipo de análisis nos permite clasificar y agrupar datos en función de sus características. En el caso de este proyecto, se puede utilizar para recoger relaciones entre los datos que no son visibles en un análisis general.

Existen numerosas herramientas de minería de datos con la funcionalidad que se necesita en este proyecto. Las más conocidas y utilizadas las detallaremos a continuación:

Weka

Weka [WKA] es una colección de algoritmos de inteligencia artificial diseñada para tareas de minería de datos. Esta plataforma contiene herramientas para pre-procesado de datos, clasificación, agrupamiento, regresión, reglas de asociación y visualización. Esta herramienta está desarrollada en **Java** y fue diseñada por la Universidad de Waikato de Nueva Zelanda y su nombre y logo proviene de un ave perteneciente a la fauna neozelandesa.

Figura 3: Logotipo de la herramienta Weka

A continuación se enumeraran las ventajas e inconvenientes de la herramienta:

- Ventajas:
  - Permite la ejecución por medio de mandatos
  - Está disponible bajo la licencia publica general GNU
  - Portabilidad
  - Multiplataforma
  - Extensa colección de técnicas de preprocesado y modelado

- Inconvenientes:
  - Poca documentación
  - No cubre el modelado en secuencia

RapidMiner

RapidMiner [RMN] es un programa informático para análisis y minería de datos. Permite realizar procesos de análisis a través de su interfaz. En el año 2010, en una encuesta realizada por KDnuggets, una prestigiosa página de minería de datos, le sitúa en el número uno del ranking de plataformas más utilizadas. Desarrollada por el departamento de Inteligencia Artificial por la universidad de Dortmund, se distribuye bajo licencia AGPL.

Figura 4: Logotipo de la herramienta RapidMiner

A continuación se enumeraran las ventajas e inconvenientes de la herramienta:

- Ventajas:
  - Desarrollado en Java
  - Multiplataforma
  - Accesible a través de bibliotecas propias
  - Permite el desarrollo a través de scripts

- Inconvenientes:
  - Dificultad de uso
  - Uso excesivo de memoria

En definitiva, se puede catalogar RapidMiner como la plataforma de datamining más completa y accesible debido a la infinidad de canales con los que se puede acceder a las herramientas contenidas en la aplicación. Y Weka como una herramienta más sencilla altamente portable con una extensa colección de técnicas de modelado y procesado de datos. En el apartado de elección de la tecnología de desarrollo evaluaremos que tecnología se adapta mejor al proyecto especificado en este documento.

2.3 Magento y el modelo de datos EAV

En este apartado se expondrán varias plataformas de comercio electrónico haciendo una comparativa con la plataforma utilizada en este proyecto, Magento. Se explicaran los puntos fundamentales de la plataforma para establecer un marco tecnológico.

Magento y otras plataformas e-commerce

Magento es una plataforma de contenidos web para e-commerce o comercio electrónico de código abierto. Entre sus principales características están su notable flexibilidad y escalabilidad, lo que permite poder desarrollar prácticamente cualquier tipo de proyecto de comercio electrónico.

Según un estudio realizado sobre el "*Top 1 milion*" del sitio web Alexa, los resultados de la plataforma para desarrollar portales de comercio electrónico, destacan Magento con un 30% como la plataforma más utilizada.

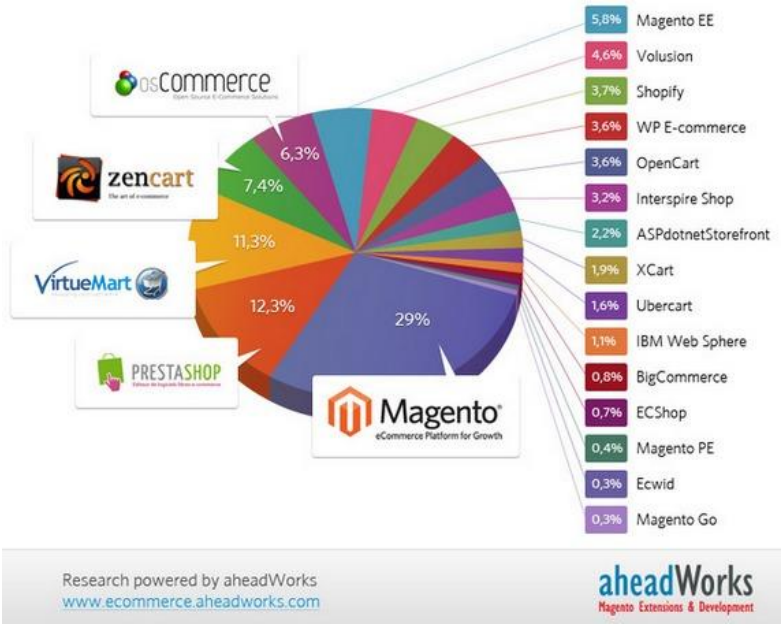


Figura 5: Propagación de las plataformas e-commerce en Julio de 2013

A continuación realizamos un breve comparativa entre Magento y Prestashop, la siguiente plataforma más utilizada, y obtendremos las diferencias fundamentales entre ellas.

Prestashop

Prestashop [PRS] es una plataforma de comercio electrónico sencilla de instalar y configurar indicada principalmente para PYMES debido principalmente a su buena indexación con los buscadores (buen posicionamiento web), soporte multimedia e interfaz agradable.
A continuación veremos los aspectos más destacados y en contrapartida, las deficiencias de esta plataforma:

- Ventajas:
  - Coste final de un proyecto e-commerce profesional económico
  - Fácil de utilizar
  - Soporte para códigos de barras
  - Panel de administración intuitivo
  - Comunidad de usuarios muy grande
  - Bajo consumo en recursos
- Inconvenientes:
  - Soporte mayoritariamente en inglés y francés
  - Sistema deficiente de atributos
  - Configuración insuficiente para proyectos grandes

Magento

Magento [MGN] es una plataforma de comercio electrónico indicada para cualquier tipo de proyecto con un alto grado de personalización. A continuación veremos los aspectos más destacados y en contrapartida, las deficiencias de esta plataforma:

- Ventajas:
  - Muy potente
  - Se puede personalizar prácticamente todo
  - Permite gestionar varias tiendas
  - Sistema de búsquedas en Ajax, lo que permite realizar búsquedas sin comprometer la eficiencia del sistema
  - Soporte para códigos de barras
  - Panel de administración muy completo
  - Gestión de pedidos
- Inconvenientes:
  - Comunidad de usuarios pequeña
  - Costes de desarrollo altos
  - Panel de administración complejo
  - Curva de aprendizaje muy alta

En definitiva, se puede catalogar a la plataforma Magento como la plataforma e-commerce más flexible, tanto en ámbitos de configuración como de gestión de datos, y completa debido a que sirve para cualquier tipo de tienda o sistema de tiendas.
Una de las cosas que hace tan flexible a Magento es el modelo de datos. Magento cuenta con un modelo de datos EAV, lo que le permite ser infinitamente más escalable que al modelo tradicional de Prestashop. En los enlaces expuestos a continuación se puede verificar la complejidad de ambos modelos:

Esquema de datos de Prestashop completo

Modelo básico de datos de Magento

A continuación se explicarán ambos modelos con el fin de aclarar la diferencia de escalabilidad entre ambos.

Modelo de datos EAV

La plataforma de comercio electrónico Magento cuenta con un modelo de datos que difiere considerablemente del modelo relacional tradicional. La plataforma cuenta con un modelo de datos EAV o modelo Entidad-Atributo-Valor. El modelo de datos EAV [EAV], cuenta con una serie de características, entre las que se encuentra como principal su **escalabilidad**. En el modelo tradicional, se define una entidad y se definen los atributos que contiene esa entidad. Por ejemplo la entidad *person* cuenta con los atributos: *dni, name, surname, birthdate*, etc. Todos los atributos tienen un tipo de dato determinado; y la representación en la base de datos sería la tabla *persona*, con los atributos que hemos definido anteriormente.

Persona				
dni	name	surname	addreess	children
46512348Y	Juan	Lopez	c\maiz	0
75121325K	Ana	Sanz	c\heno	2
45645774R	Bob	Smith	c\mayor	1
11326258M	Manuel	Gasol	c\real	0

Tabla 1: Tabla Persona. Modelo Relacional Tradicional

El modelo EAV es un poco más complejo. En este modelo los valores de los atributos de las entidades no van definidos directamente, es decir, se definen de forma independiente. En este modelo, se definen entidades que se relacionan con otras entidades donde se definirían los atributos o campos del modelo tradicional; y a su vez estas se relacionan con otras entidades

donde están contenidos los valores.

Este modelo cuenta con 3 entidades fundamentales:

- **Entidades:** hacen referencia a cada una de las clases que componen el modelo de datos. En el modelo tradicional sería el modelo o tabla.

- **Atributos:** hacen referencia a cada uno de los campos que constituyen la definición de los modelos. Las características de cada entidad

- **Valores:** representan a la unidad mínima de información en un modelo de datos. Constituyen la información, de un atributo contenido en una entidad.

A continuación se realizará un recorrido por el modelo EAV utilizando el mismo ejemlpo que en el apartado anterior.

En el modelo EAV se define una entidad *person* en la tabla de entidades. Esta tabla posee el prefijo "eav_" debido a que es una clase principal y predefinida de este modelo.

eav_entity_type		
entity_type_id	entity_type_code	entity_table
1	person	person/entity
2	product	catalog/product

Tabla 2: Tabla de definición de entidades. Modelo EAV

Como podemos observar en la Tabla 2, en el campo *entity*_table, se define dónde se va a almacenar cada una de las instancias de esa entidad. Este campo marca la **pauta principal** a la hora de nombrar cada una de las entidades relacionadas con el *entity_type_id* = 1; es decir, tipo *person.*

person_entity		
entity_id	entity_type_id	dni
1	1	46512348Y
2	1	75121325K
3	1	45645774R
4	1	11326258M

Tabla 3: Tabla de la entidad person. Modelo EAV

Dentro de esa tabla *person_entity*, encontramos varios campos interesantes:

- **entity_id**: contiene la clave única de instancia y hace referencia a cada uno de los *person* que contiene nuestro modelo.

- **entity_type_id:** hace referencia a la entidad principal a la que pertenece la instancia. Si comprobamos en la tabla *eav_entity_type* el mismo campo, veremos que coincide con el *entity_type_code* = person.

- **dni:** contiene el identificador de la instancia (actuaría como clave primaria en el modelo tradicional)

Ligadas a la Tabla 3, existen las tablas que contienen los valores de las instancias de tipo *person* definidas. Estas tablas siguen la pauta de nomenclatura definida en campo entity_table de la Tabla 2. Estas tablas son: *person_entity_int, person_entity_varchar, person_entity_datetime, person_entity_decimal, person_entity_text;* y todas ellas hacen referencia a los datos de las instancias dependiendo del tipo de dato que se almacene. A continuación se mostrarán algunos ejemplos relacionados con la entidad *person*.

person_entity_int				
value_id	entity_type_id	attribute_id	entity_id	value
1	1	4	1	0
2	1	4	2	2
3	1	4	3	1
4	1	4	4	0

Tabla 4: Tabla de la entidad person para datos de tipo entero. Modelo EAV

person_entity_varchar				
value_id	entity_type_id	attribute_id	entity_id	value
1	1	1	1	Juan
2	1	2	1	Lopez
3	1	3	1	c\maiz
4	1	1	2	Ana
5	1	2	2	Sanz
6	1	3	2	c\heno
7	1	1	3	Bob
8	1	2	3	Smith
5	1	3	3	c\real
6	1	1	4	Manuel

7	1	2	4	Gasol
8	1	3	4	c\mayor

Tabla 5: Tabla de la entidad person para datos de tipo varchar. Modelo EAV

Llegados a este punto, ya está definida la entidad donde se almacenan las instancias de *person* así como los datos que hacen referencia a esta entidad, pero ¿Cómo se relacionan cada uno de los datos contenidos en las tablas de tipo con la entidad? Por medio de los atributos.

En el modelo EAV existe una tabla que también es una clase principal del modelo (al igual que *eav_entity_type*), que es *eav_attribute*. En eav_attribute se definen los atributos independientemente de la entidad a la que pertenezcan.

eav_attribute		
attribute_id	entity_type_id	attribute_code
1	1	name
2	1	surname
3	1	address
4	1	children
5	1	birthdate
6	2	barcode
7	2	city
8	2	postal_code
9	2	tax

Tabla 6: Tabla de atributos. Modelo EAV

Como se puede observar en la Tabla 6, esta contiene todos los atributos del modelo; los de tipo *person* están definidos por en *entity_type_id* = 1, pero también están el resto de atributos del modelo que tienen relación con otras entidades. Esta clase cuenta con 3 atributos principales:

- **attribute_id:** identificador único de atributo. Con este atributo se pueden relacionar los valores contenidos en las tablas de tipo de la entidad con el descriptor del atributo.
- **entity_type_id:** hace referencia a la entidad con la que está relacionado ese atributo.
- **attribute_code:** representa el nombre del atributo.

Así quedaría definido el modelo *person* según el modelo EAV.

Magento es una plataforma de comercio electrónico **flexible** y **escalable**; y como tal, tiene que poder adaptarse a cada uno de los diferentes modelos de tiendas. Esta es la razón principal por la que **Magento utiliza un modelo de datos EAV, por la maleabilidad de este modelo.** En este modelo se pueden añadir los atributos que sean necesarios y relacionarlos con las entidades que el gestor crea conveniente. Este modelo le permite al gestor de la tienda añadir cualquier tipo de atributos a las diferentes entidades, incluso crear nuevas, debido a que los atributos de las entidades no están preestablecidos.

Uno de los principales **inconvenientes** de este modelo de datos es que computacionalmente es un **modelo muy ineficiente** a la hora de trabajar con él**.** A continuación se procederá a evaluar la eficiencia del modelo a la hora de extraer información de este modelo. Basaremos la evaluación en el modelo *person* definido anteriormente y lo compararemos con el modelo relacional Tradicional. A la hora de realizar el seguimiento utilizaremos la siguiente leyenda:

> Los datos referentes a la consulta se expresan en rojo y los resultados que devuelve la consulta en verde.

La consulta a realizar es la siguiente:

> Consiste en **extraer del modelo el name y children de la persona con dni 46512348Y**

Para extraer *name* y *children* en el Modelo Tradicional, sería suficiente con hacer una consulta de esta forma: **EXTRAER (name y children) DE *person* CUANDO (dni = 46512348Y) ***; donde el coste computacional sería mínimo debido a que los campos están definidos en la tabla y los valores contenidos en ella.

Persona				
dni	name	surname	addreess	children
46512348Y	Juan	Lopez	c\maiz	0
75121325K	Ana	Sanz	c\heno	2
45645774R	Bob	Smith	c\mayor	1
11326258M	Manuel	Gasol	c\real	0

Figura 6: Consulta. Modelo Tradicional

Para extraer *name* y *children* en el Modelo EAV, la tarea no es tan sencilla. Se explicarán cada uno de los pasos directamente de forma gráfica:

1. Se extrae el entity_type_id por medio de la nomenclatura de la entidad:

eav_entity_type		
entity_type_id	entity_type_code	entity_table
1	person	person/entity
2	product	catalog/product

Figura 7: Consulta entity_type_id. Modelo EAV

2. Con el entity_type_id extraido y el dni, extraemos el entity_id de la persona que buscamos

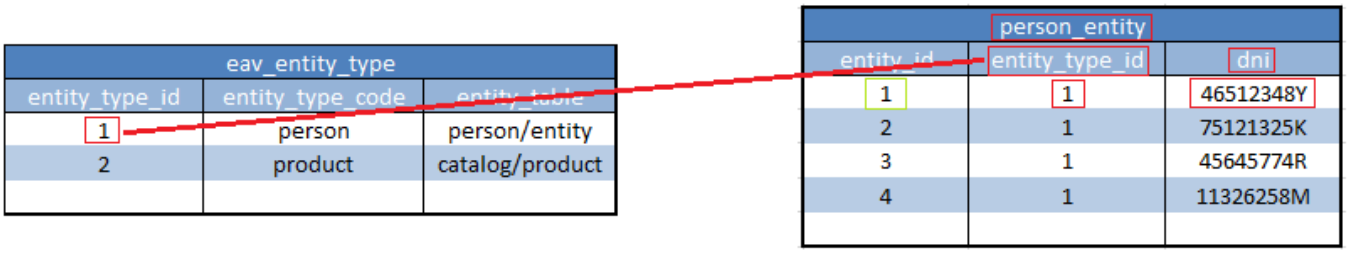
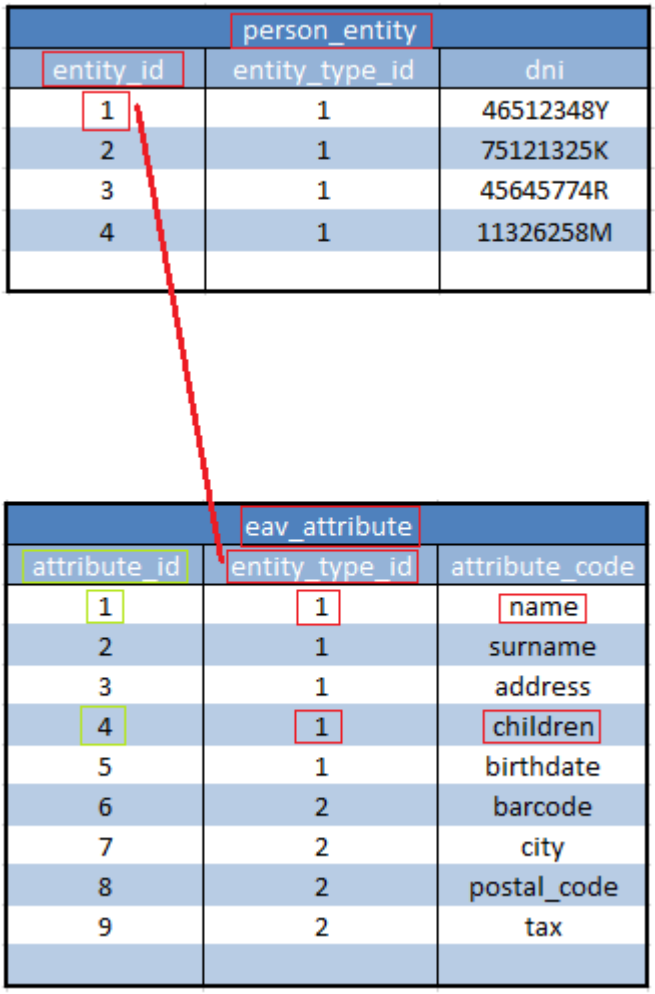


Figura 8: Consulta entity_id. Modelo EAV

3. Con el entity_id obtenido, extraemos los attribute_id de los atributos que queremos conseguir.



person_entity

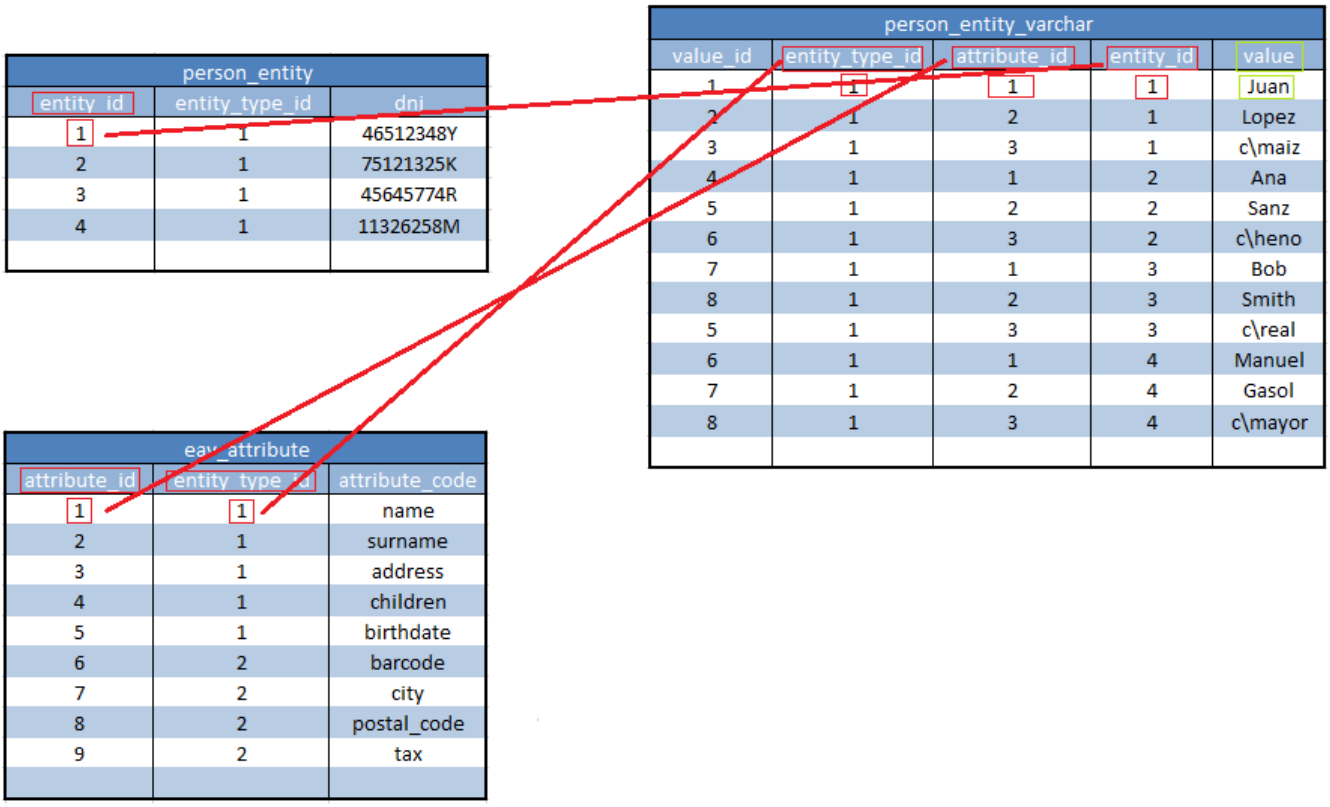
entity_id	entity_type_id	dni
1	1	46512348Y
2	1	75121325K
3	1	45645774R
4	1	11326258M

eav_attribute

attribute_id	entity_type_id	attribute_code
1	1	name
2	1	surname
3	1	address
4	1	children
5	1	birthdate
6	2	barcode
7	2	city
8	2	postal_code
9	2	tax

Figura 9: Consulta attribute_id. Modelo EAV

4. Con el attribute_id, entity_type_id y entity_id **extraemos el valor de name** en la tabla donde se almacenan los datos de tipo varchar.



person_entity

entity_id	entity_type_id	dni
1	1	46512348Y
2	1	75121325K
3	1	45645774R
4	1	11326258M

person_entity_varchar

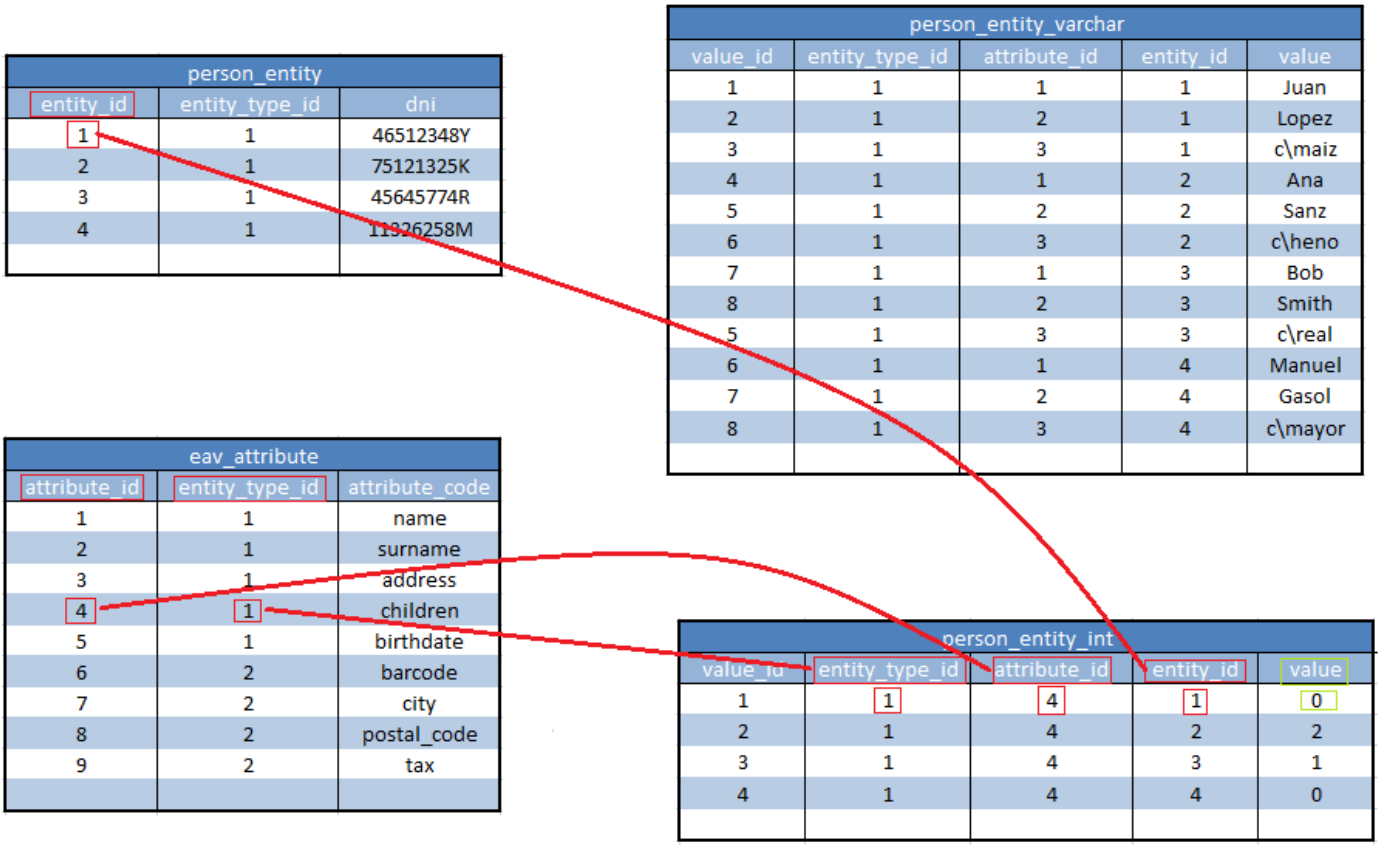
value_id	entity_type_id	attribute_id	entity_id	value
1	1	1	1	Juan
2	1	2	1	Lopez
3	1	3	1	c\maiz
4	1	1	2	Ana
5	1	2	2	Sanz
6	1	3	2	c\heno
7	1	1	3	Bob
8	1	2	3	Smith
5	1	3	3	c\real
6	1	1	4	Manuel
7	1	2	4	Gasol
8	1	3	4	c\mayor

eav_attribute

attribute_id	entity_type_id	attribute_code
1	1	name
2	1	surname
3	1	address
4	1	children
5	1	birthdate
6	2	barcode
7	2	city
8	2	postal_code
9	2	tax

Figura 10: Consulta person_entity_varchar. Modelo EAV

5. Con el attribute_id, entity_type_id y entity_id **extraemos el valor de children** (número de hijos) en la tabla donde se almacenan los datos de tipo int.



person_entity

entity_id	entity_type_id	dni
1	1	46512348Y
2	1	75121325K
3	1	45645774R
4	1	11326258M

person_entity_varchar

value_id	entity_type_id	attribute_id	entity_id	value
1	1	1	1	Juan
2	1	2	1	Lopez
3	1	3	1	c\maiz
4	1	1	2	Ana
5	1	2	2	Sanz
6	1	3	2	c\heno
7	1	1	3	Bob
8	1	2	3	Smith
5	1	3	3	c\real
6	1	1	4	Manuel
7	1	2	4	Gasol
8	1	3	4	c\mayor

eav_attribute

attribute_id	entity_type_id	attribute_code
1	1	name
2	1	surname
3	1	address
4	1	children
5	1	birthdate
6	2	barcode
7	2	city
8	2	postal_code
9	2	tax

person_entity_int

value_id	entity_type_id	attribute_id	entity_id	value
1	1	4	1	0
2	1	4	2	2
3	1	4	3	1
4	1	4	4	0

Figura 11: Consulta person_entity_int. Modelo EAV

Como se puede observar, hemos tenido que realizar 5 pasos antes de obtener los 2 datos. Con el modelo tradicional, obteníamos ambos datos con 1 solo paso y sin tener que relacionar tablas. Teniendo en cuenta que esto es un ejemplo muy simplificado del modelo real de una tienda, **se puede concluir que el modelo EAV es muy flexible, pero en contrapartida es complejo y costoso computacionalmente manejar los datos.**

2.4 Elección de la tecnología de desarrollo

El objetivo de este apartado consiste en decidir la tecnología que se va a utilizar para desarrollar la solución. El objetivo principal del trabajo, como ya se ha especificado anteriormente, reside en desarrollar una aplicación cliente servidor para realizar análisis predictivo sobre los datos recogidos en la plataforma Magento. Por tanto, la tecnología seleccionada tiene que proveer a la aplicación la capacidad de generar análisis predictivo con modelos de aprendizaje supervisado y no supervisado. Cabe destacar, que al ser una aplicación cliente servidor, será necesario evaluar las diferentes tecnologías de desarrollo para aplicaciones web. Éstas nos proporcionarán un elemento sobre el que integrar a cada uno de los componentes mencionados anteriormente. Por medio de esta plataforma web, obtendremos una plataforma robusta sobre la que realizar el análisis predictivo, recoger los datos de Magento y proporcionar un interfaz al usuario para poder interactuar con los modelos contenidos en la plataforma de minería de datos elegida.

La estructura del apartado será la siguiente. En primer lugar se generarán apartados para analizar cada una de las disciplinas a cubrir por el proyecto. En segundo lugar se definirán los requisitos a cubrir por la plataforma seleccionada y a continuación se expondrán en cada una de las tecnologías a estudiar para cada una de las necesidades a cubrir.

Disciplinas a cubrir por el proyecto:

- Tecnología de desarrollo de la aplicación cliente-servidor
- Tecnología para generación de modelos

Tecnología de desarrollo de la aplicación cliente-servidor

En este apartado se analizarán cada una de las alternativas para desarrollar la aplicación cliente-servidor en la que se basa el proyecto

Requisitos a cubrir por la tecnología

A continuación se establecen una serie de requisitos fundamentales que han de ser cubiertos, en la medida de lo posible, por la tecnología seleccionada:

- **Multiplataforma:** la tecnología seleccionada tiene que permitir un despliegue independiente de la plataforma evitando así, las restricciones a la hora de la implementación. Al tratarse de una tecnología web, se busca que el despliegue de la aplicación no dependa del navegador web en el que se ejecuta. Se tendrá en alta consideración la independencia completa del navegador web en el que se ejecute.

- **Capacidad de integración de las herramientas de análisis predictivo:** la tecnología que se seleccione debe de proporcionar la posibilidad de integrar la tecnología para generación de modelos de la manera más sencilla posible. Todas las tecnologías de generación de modelos que se van a evaluar están desarrolladas sobre el lenguaje de programación **Java.** Si se diera el caso de que una de las tecnologías a analizar no pudiese integrar dicha plataforma de generación de modelos, ésta se descartará inmediatamente. Se tendrá en alta consideración el hecho de que se pueda integrar la herramienta de la forma más sencilla posible para su elección.

- **Integración de datos sobre el modelo EAV:** la tecnología que se seleccione, debe tener la capacidad suficiente para trabajar con el modelo de datos EAV. Debido a la complejidad de este modelo de datos, se tendrá en alta consideración el hecho de que el manejo de los modelos de datos sea lo más independiente y sencilla de utilizar.

- **Escalabilidad:** la tecnología que se seleccione deberá tener la posibilidad de modificar los modelos de datos importados de Magento. Esta funcionalidad se espera del sistema debido a que como Magento es altamente flexible, es deseable que la aplicación tenga la posibilidad de adaptar el modelado de datos a las necesidades del usuario.

- **Material de ayuda y referencias (APIs y librerías):** la tecnología que se seleccione deberá tener un volumen adecuado de material de ayuda y referencias. No sólo debe tener un volumen adecuado, sino que las referencias sean sencillas de entender y de calidad. Se tendrá en alta consideración que existan tanto fuentes oficiales (libros, artículos de investigación y demás publicaciones oficiales) como fuentes no oficiales como foros o páginas web en las que se pueda consultar la información.

- **Curva de aprendizaje:** la tecnología que se seleccione debe de facilitar el tiempo de aprendizaje de los lenguajes de programación utilizados. También es importante el tiempo necesario para familiarizarse con las herramientas complementarias y el entorno de desarrollo. Se tendrá en alta consideración la sencillez y el menor tiempo posible de aprendizaje.

- **Licencia de software:** se tendrá en alta consideración que la tecnología sea de libre acceso para usuarios tanto para los desarrolladores como para los usuarios finales.

Principales características de las tecnologías

En este apartado se describen las principales características de las tecnologías que se van a analizar. Se acompañarán de una tabla en el que se indica cómo cubren cada uno de los requisitos definidos en el apartado anterior.

Ruby on Rails

RubyOnRails [ROR] es un framework de aplicaciones web escrito en el lenguaje de programación Ruby, siguiendo la arquitectura un Modelo Vista Controlador (MVC). RoR es un framework de código abierto y que trata de combinar la potencia de las aplicaciones web tradicionales con un lenguaje que posee una sintaxis muy legible. Este framework, posee numerosos módulos o gemas a modo de complementos que actúan como bibliotecas para desarrollar de forma sencilla módulos ya implementados. Esta forma de distribución, está desarrollada bajo la filosofía principal de Ruby on Rails, "no te repitas" (del inglés, **Don't Repeat Yourself**).

Multiplataforma	Compatible con los navegadores de Windows, Linux y Mac OS.
Escalabilidad	Ofrece la posibilidad de generar migraciones, es decir, modificaciones estructurales del modelo de datos sin modificar el contenido. Estas modificaciones son reversibles.
Capacidad de integración de las herramientas de análisis predictivo	Contiene un módulo un o gema llamada 'rjb' que permite usar java con una interfaz nativa
Integración de datos sobre el modelo EAV	Utiliza el modelo EAV, lo que permite un trato independiente de los datos y permite tratar los datos de forma muy intuitiva
Material de ayuda y referencias	Amplio. Numerosas fuentes oficiales y no oficiales. Es una tecnología de desarrollo con gran repercusión en los últimos años.
Curva de aprendizaje	No se cuenta con experiencia previa ni en el entorno de desarrollo, ni en los lenguajes de programación, ni en patrón de arquitectura MVC. La curva de aprendizaje cuenta un una gran pendiente.

Licencia de software	Libre y gratuita

Figura 12: Requisitos cubiertos por Ruby on Rails

Adobe Flex

Abobe Flex [AFX] es un framework que agrupa varias tecnologías para dar soporte al despliegue y desarrollo de aplicaciones enriquecidas de Internet basadas en la plataforma Flash. Proporciona un lenguaje moderno basado en estándares y el modelo de programación que soporta patrones de diseño común adecuado para los desarrolladores de diferentes orígenes.

Multiplataforma	Compatible con los navegadores de Windows, Linux y Mac OS. No compatible con dispositivos Apple.
Escalabilidad	Actúa bajo el modelo de datos relacional tradicional. Lo que vuelve el sistema **poco escalable.**
Capacidad de integración de las herramientas de análisis predictivo	Integración de java por medio de BlazeDS
Integración de datos sobre el modelo EAV	Actúa bajo el modelo de datos relacional tradicional. Obliga a mapear el modelo EAV a un modelo tradicional. Modelos no son independientes del resto de componentes de la aplicación.
Material de ayuda y referencias	Amplio. Numerosas fuentes oficiales y no oficiales fruto de un largo recorrido en el mercado.
Curva de aprendizaje	Se cuenta con experiencia previa en esta tecnología. La curva de aprendizaje es prácticamente inexistente.
Licencia de software	Libre y gratuita

Figura 13: Requisitos cubiertos por Adobe Flex

Tecnología de desarrollo seleccionada

Teniendo en cuenta lo evaluado en el apartado anterior, la tecnología seleccionada para desarrollar la aplicación cliente-servidor es **RubyOnRails**. El motivo fundamental por el que se ha seleccionado Ruby on Rails como tecnología para desarrollar la aplicación es la combinación de la arquitectura Modelo vista controlador y la posibilidad de generar migraciones. Uno de los problemas fundamentales del problema a abordar, es el tratamiento de los datos; el modelo de datos EAV no es un modelo muy utilizado por lo que no es habitual encontrar herramientas que trabajen sobre este modelo. Aunque Ruby on Rails trabaje con un modelo relacional tradicional, brinda la posibilidad de generar modificaciones sobre los modelos de datos una vez generados, es decir, que si el gestor de la plataforma Magento decide añadir nuevos atributos sobre las entidades del modelo, se podría realizar una modificación reversible sobre el modelo de datos de nuestra aplicación. Además de poder realizar estas modificaciones, la arquitectura MVC nos permite tratar los modelos de manera y dependiente a los interfaces o vistas, y de los controladores.

Con respecto a Adobe Flex, aparte de no ofrecer compatibilidad con todos los navegadores, el principal motivo por el que se descarta esta tecnología es debido a que no sólo es necesario mapear el modelo de datos, sino que no ofrece la posibilidad de modificar el modelo de forma independiente.

Tecnología para generación de modelos

En este apartado se analizarán cada una de las alternativas para generar los modelos que se utilizaran en el análisis predictivo.

Requisitos a cubrir por la tecnología

A continuación se establecen una serie de requisitos fundamentales que han de ser cubiertos, en la medida de lo posible, por la tecnología seleccionada:

- **Capacidad de integración:** es deseable que la herramienta seleccionada sea sencilla de integrar en la aplicación y si fuera necesario, en el sistema. Se tendrá en consideración la independencia del lenguaje y sistema operativo en el que se ejecute.

- **Flexibilidad:** la herramienta que se seleccione deberá tener la posibilidad de trabajar con diferentes tipos y volúmenes de datos, es deseable que la aplicación tenga la posibilidad de adaptar el modelado de datos introducidos.

- **Material de ayuda y referencias (APIs y librerías):** la herramienta que se seleccione deberá tener un volumen adecuado de material de ayuda y referencias. No sólo debe tener un volumen adecuado, sino que las referencias sean sencillas de entender y de calidad. Se tendrá en alta consideración que existan tanto fuentes oficiales (libros, artículos de investigación y

demás publicaciones oficiales) como fuentes no oficiales como foros o páginas web en las que se pueda consultar la información.

- **Curva de aprendizaje:** la herramienta que se seleccione debe de facilitar el tiempo de aprendizaje del funcionamiento de la misma. También es importante el tiempo necesario para familiarizarse con las herramientas complementarias y el entorno de desarrollo. Se tendrá en alta consideración la sencillez y el menor tiempo posible de aprendizaje.

- **Licencia de software:** se tendrá en alta consideración que la tecnología sea de libre acceso para usuarios tanto para los desarrolladores como para los usuarios finales.

Principales características de las herramientas

En este apartado se describen las principales características de las herramientas que se va a analizar. Se acompañarán de una tabla en el que se indica cómo cubren cada uno de los requisitos definidos en el apartado anterior.

RapidMiner

RapidMiner un es un programa informático para el análisis y minería de datos. Permite desarrollar procesos de análisis de datos a través del encadenamiento de operadores. Es la herramienta de minería de datos más utilizada actualmente y fue desarrollado por la Universidad de Dortmund en 2001.

Capacidad de integración	Línea de comandos, *batches* o lotes y bibliotecas.
Flexibilidad	Trabaja con múltiples tipos de datos y está preparado para todo tipo de conjuntos de datos o DataSets
Material de ayuda y referencias	Amplio. Numerosas fuentes oficiales y no oficiales. Es la herramienta más utilizada de minería de datos
Curva de aprendizaje	No se cuenta con experiencia previa ni en el entorno. La curva de aprendizaje cuenta con una gran pendiente.
Licencia de software	Libre y gratuita (LGPL)

Figura 14: Requisitos cubiertos por RapidMiner

Weka

Weka es una colección de algoritmos de inteligencia artificial diseñada para tareas de minería de datos. Esta plataforma desarrollada en **Java** contiene herramientas para pre-procesado de datos, clasificación, agrupamiento, regresión reglas de asociación y visualización.

Capacidad de integración	Línea de comandos y biblioteca.
Flexibilidad	Trabaja con múltiples tipos de datos y está preparado para todo tipo de conjuntos de datos o DataSets
Material de ayuda y referencias	Suficiente. API de Weka y fuentes no oficiales.
Curva de aprendizaje	Experiencia previa en diversas asignaturas de la mención en Computación. La curva de aprendizaje cuenta un una leve pendiente.
Licencia de software	Libre y gratuita (GPL)

Figura 15: Requisitos cubiertos por Weka

Herramienta de generación de modelos seleccionada

Teniendo en cuenta lo evaluado en el apartado anterior, la herramienta seleccionada para generar los modelos es **Weka**. El motivo fundamental por el que se ha seleccionado Weka es porque se cuenta con una experiencia previa en la herramienta que puede simplificar y facilitar mucho el desarrollo del proyecto.

Con respecto a RapidMiner, cuenta con mayor número de referencias y material de ayuda y la capacidad integración es incluso mejor que en Weka. El problema está en que, aunque sea una herramienta más completa, las tareas de modelado que se van a utilizar están contenidas en ambas herramientas y por tanto es más sencillo partir de una herramienta conocida.

3 Gestión de proyecto software

El objetivo de este apartado es realizar una simulación de la gestión del proyecto el software desarrollado. Consiste en una estimación hipotética del proyecto como si fuese a desarrollar con las condiciones habituales de un entorno empresarial. El proyecto ha tenido una duración de 8 meses, desde Enero hasta Septiembre de 2014, ambos incluidos. En los siguientes apartados se incluye la estimación de las tareas y recursos empleados así como el presupuesto que se estima para desarrollar dicho proyecto.

3.1 Estimación de tareas y recursos

En este apartado se presentan los costes asociados al desarrollo del proyecto. Se tienen en cuenta los costes de personal, costes de hardware y de software.

Costes de personal

Los costes de personal los constituyen los salarios del equipo encargado de desarrollar el proyecto. En primer lugar, se definirán los roles de los empleados:

- **Jefe de proyecto**: es la persona que administra y controla los recursos asignados a un proyecto, con el propósito de que se cumplan correctamente los planes definidos.
- **Analista**: es la persona que trabaja con el cliente para realizar el análisis y especificación del sistema, es decir, dividir el problema a resolver en sub-problemas de menos complejidad.
- **Diseñador**: es la persona encargada de generar el diseño arquitectónico del sistema, el diseño detallado y los prototipos del sistema.
- **Programador**: es la persona encargada de convertir la especificación del sistema en código fuente ejecutable utilizando uno o más lenguajes de programación, así como herramientas de software de apoyo a la programación.

El equipo de desarrollo del proyecto contará con un empleado por cada uno de los roles definidos anteriormente. El salario dependerá del rol que desempeñe el empleado así como la jornada de trabajo. El jefe de proyecto trabajará a media jornada durante todos los días de la semana en este proyecto así como el analista, debido a que sus tareas son requeridas en momentos puntuales del

proyecto; el diseñador y el programador trabajarán una jornada completa diaria hasta la finalización del proyecto. El salario bruto mensual se calculará de la siguiente forma:

$$Salario\ bruto\ mensual = \frac{coste}{hora} \times \frac{horas}{día} \times \frac{días}{mes}$$

A continuación se mostrará la tabla salarial del equipo de trabajo:

Empleado	coste/hora (€)	jornada	dias/mes	salario bruto mensual (€)
Jefe de protyecto	20	media	7	560
Analista	15	media	10	600
Diseñador	8	entera	15	960
Programador	7	entera	17	952
TOTAL (€)				3072

Tabla 7: Salario bruto mensual del equipo de trabajo

A continuación se muestran los costes asociados a los ocho meses del proyecto, teniendo en cuenta la cuota de la seguridad social, cuyo porcentaje es del 23,6% según las bases y tipos de cotización [BTC] establecido por el Ministerio de Empleo y Seguridad Social. Se calculará el coste total de cada empleado en el proyecto siguiendo esta fórmula:

$$Coste\ total = (salario\ bruto\ mensual + cuota\ SS) \times meses$$

A continuación se mostrarán los costes totales de personal:

Empleado	Salario bruto mensual (€)	Cuota Seguridad Social	Meses de trabajo	Coste total (€)
Jefe de protyecto	560	132,16	10	6921,6
Analista	600	141,6	4	2966,4
Diseñador	960	226,56	7	8305,92
Programador	952	224,672	7	8236,704
TOTAL (€)				26430,624

Tabla 8: Costes totales de personal

Costes Hardware y Software

Los costes de hardware y software, incluyen aquellos materiales de este tipo asociados al proyecto a desarrollar.

Recursos hardware utilizados han sido los siguientes:

- Ordenador de sobremesa Asus CG8270
- 3 Ordenadores portátiles Asus Zenbook UX301LA
- Portatil MacBook Air
- Monitor Samsung SyncMaster T240

A continuación se mostrarán los costes totales de hardware:

Recurso	Coste (€)	Unidades	Coste total (€)
Asus CG8270	1055	1	1055
Asus Zenbook UX301LA	1870	1	1870
MacBook Air	929	1	929
Samsung T240	139	1	139
TOTAL (€)			3993

Tabla 9: Costes hardware

Recursos software utilizados han sido los siguientes:

- Microsoft Visio 2010
- Microsoft Word 2013
- Microsoft Excel 2013

A continuación se mostrarán los costes totales de software:

Recurso	Coste (€)	Unidades	Coste total (€)
Microsoft Visio 2010	99	1	99
Microsoft Word 2013	249	1	249
Microsoft Excel 2013			
TOTAL (€)			348

Tabla 10: Costes software

3.2 Presupuesto

Una vez realizado el desglose de los costes de personal y de los recursos software y hardware, el coste asociado al proyecto es el que se muestra en la siguiente tabla:

Concepto	Coste (€)
Costes de personal	26430,62
Costes hardware	3993
Costes software	348
Total (€)	30771,62

Tabla 11: Costes totales

Para concluir, se presenta en la tabla 12, el presupuesto final incluyendo riesgos, beneficios e IVA:

Concepto	Coste (€)
Costes totales	30771,62
Riesgos (15%)	4615,74
Riesgos (20%)	6154,32
TOTAL (sin IVA)	41541,68
IVA (21%)	8723,75
Total (€)	50265,43

Tabla 12: Presupuesto final

El presupuesto total del proyecto expresado en el presente documento asciende a 50.265,43€ (CINCUENTA MIL DOSCIENTOS SESENTA Y CINCO EUROS CON CUARENTA Y TRES CENTIMOS)

Leganés a 24 de Septiembre de 2014

El ingeniero proyectista

Fdo. Daniel Lozano Cofrades

3.3 Plan de trabajo

En este apartado se especifica el plan de trabajo que se ha determinado para el desarrollo del proyecto. Se expondrán la clasificación de tareas así como su planificación temporal a lo largo del recorrido del proyecto.

Identificación de tareas

A continuación se especifican las tareas que componen el proyecto:

- **Estado del arte:** se define el problema a resolver por el proyecto y se establece un contexto en el que se sitúa. En última instancia, se realiza una comparativa entre las tecnologías y herramientas analizadas para escoger las más adecuadas.

- **Análisis del problema:** se describen las principales características del problema a resolver por medio de la identificación de **requisitos**.

- **Modelado de datos:** se estudiará el modelo de datos de la plataforma Magento para generar un modelo de datos adaptable a la plataforma que se va a implementar.

- **Diseño de la interfaz:** se define la interfaz del sistema, especificando todos los elementos (transiciones, interfaces, componentes) que se desarrollan en el entorno.

- **Implementación. Fase 1:** esta fase hace referencia a la implementación del modelo de datos. **S**e desarrollan los componentes referentes al modelo de datos estudiados en la fase de Modelado de datos, es decir, los conjuntos de datos que se recogen de la plataforma Magento.

- **Implementación. Fase 2:** esta fase hace referencia a la implementación de los componentes visuales. Se desarrollan los componentes referentes a la interfaz de usuario.

- **Implementación Fase 3:** esta fase hace referencia a la implementación de la funcionalidad de la aplicación. Se realiza la integración de los componentes visuales con el modelo de datos.

- **Implementación Fase 4:** esta fase hace referencia a la integración de la herramienta de generación de modelos con la Fase 3, para cerrar una versión funcional de la aplicación.

- **Implementación. Fase 5:** esta fase hace referencia a la implementación de modificadores en la aplicación para realizar el preprocesado de los datos y aportar opciones a la hora de generar los modelos.

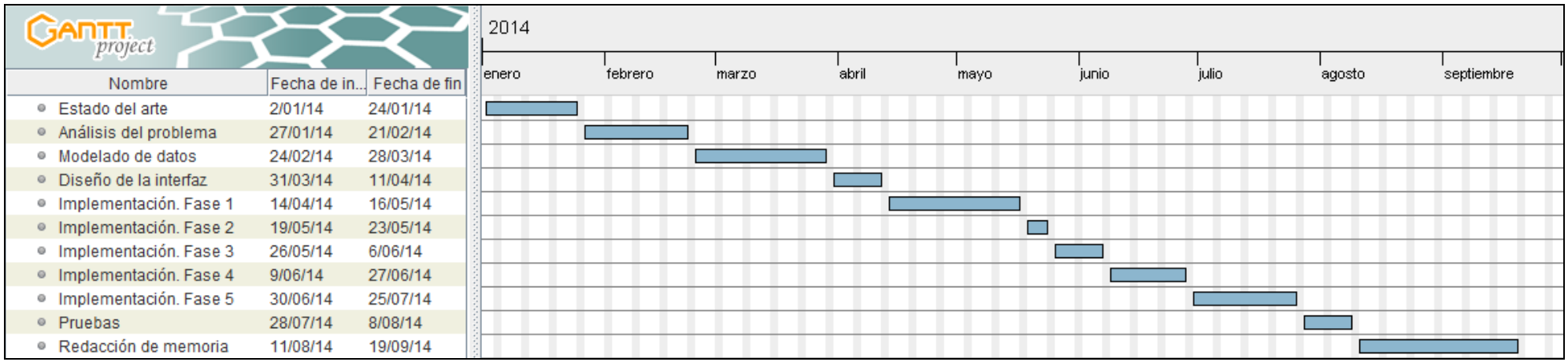
- **Pruebas:** esta fase corresponde con las pruebas que se realizan sobre la aplicación siguiendo el plan de pruebas especificado.

- **Redacción de memoria:** en esta fase se elabora el presente documento, correspondiente a la documentación del proyecto completo.

Planificación de tareas

La planificación del presente trabajo fin de grado se ha realizado por medio de un Diagrama de Gantt en el que se especifican las tareas realizadas durante el proyecto y la fecha de inicio y fin de cada una de las mismas. La planificación se ha realizado teniendo en cuenta la ejecución completa del proyecto, llevando a cabo las fases indicadas en el apartado anterior.

A continuación se muestra la planificación expresada con el diagrama de Gantt:



Nombre	Fecha de in...	Fecha de fin
Estado del arte	2/01/14	24/01/14
Análisis del problema	27/01/14	21/02/14
Modelado de datos	24/02/14	28/03/14
Diseño de la interfaz	31/03/14	11/04/14
Implementación. Fase 1	14/04/14	16/05/14
Implementación. Fase 2	19/05/14	23/05/14
Implementación. Fase 3	26/05/14	6/06/14
Implementación. Fase 4	9/06/14	27/06/14
Implementación. Fase 5	30/06/14	25/07/14
Pruebas	28/07/14	8/08/14
Redacción de memoria	11/08/14	19/09/14

Figura 16: Diagrama de Gantt

Planificación final

El presente trabajo ha cumplido la planificación inicial, habiendo finalizado todas las etapas en el tiempo estimado y habiendo sido entregado en fecha, cumpliendo los objetivos propuestos.

3.4 Gestión de riesgos

En este apartado, se expondrán y analizarán los diferentes riesgos ligados al desarrollo del trabajo.

Identificación de riesgos

A continuación se especificarán los riesgos asociados a este proyecto, siendo estos, relevantes para el desarrollo de este proyecto concreto:

- Recursos no disponibles
- Planificación demasiado optimista
- Problema en la construcción del modelo de datos
- Estabilidad de la interfaz
- Diseño inadecuado (hay que realizar un nuevo diseño)
- Análisis inadecuado del modelo de datos (hay que remodelar)
- Documentación insuficiente

Análisis de riesgos

En este apartado analizaremos la probabilidad y la magnitud de la pérdida en semanas en el caso sé que ocurriera cada uno de los riesgos especificados en el apartado anterior:

Riesgo	Probabilidad	Magnitud de perdida (semanas)	Exposición a riesgo (semanas)
Recursos no disponibles	10%	1	0,1
Planificación demasiado optimista	10%	4	0,4
Problema en la construcción del modelo de datos	15%	4	0,6
Estabilidad de la interfaz	5%	2	0,1
Diseño inadecuado (hay que realizar un nuevo diseño)	10%	2	0,2
Análisis inadecuado del modelo de datos (hay que remodelar)	20%	3	0,6
Documentación insuficiente	10%	1	0,1

Tabla 13: Análisis de riesgos

3.5 Plan de pruebas

Definición de pruebas

Los niveles de pruebas diseñados para el desarrollo de la aplicación se encuentran divididos en distintos grupos con la finalidad de controlar cada una de las partes del sistema de forma tanto individual como global.

Pruebas Unitarias

- **Fase en la que se realiza**: Construcción.
- **Personal implicado**: Equipo de desarrollo.
- **Descripción**: Este tipo de pruebas tienen como finalidad comprobar el correcto funcionamiento de cada uno de los componentes o subsistemas de forma individual. Es necesario realizar una prueba para cada función del sistema de forma independiente.
- **Tipos**: Pruebas de caja blanca y pruebas de caja negra.

En este caso se realizaran pruebas de **Caja Negra**; estas pruebas se realizan sobre las funciones del componente sin tener conocimiento del funcionamiento interno del mismo. Por este motivo, este tipo de pruebas comprueban las entradas y salidas o respuestas que produce la función.

Pruebas de Integración

- **Fase en la que se realiza:** Desarrollo.
- **Personal implicado:** Equipo de desarrollo.
- **Descripción:** Este tipo de pruebas tienen como finalidad comprobar el correcto funcionamiento de los componentes o subsistemas asumiendo el funcionamiento conjunto de las distintas partes desarrolladas.
- **Tipos:** Pruebas estructurales de integración y pruebas funcionales de integración.

Pruebas del Sistema

- **Fase en la que se realiza**: Desarrollo e implantación.
- **Personal implicado**: Equipo de desarrollo y equipo de implantación.
- **Descripción**: Este tipo de pruebas tienen como finalidad comprobar el correcto funcionamiento del sistema de forma global.
- **Tipos**: Pruebas funcionales, de comunicaciones, de rendimiento, de volumen, de sobrecarga, de disponibilidad de datos, de facilidad de uso, de operación, de entorno y de seguridad.

1. Pruebas Funcionales

   Tipo de pruebas orientadas a que el sistema cumpla con los requisitos que se han detallado en las especificaciones del mismo.

2. Pruebas de Rendimiento

   Pruebas que se realizan sobre el sistema en las que el tiempo de respuesta del mismo debe cumplir con lo especificado en el análisis y diseño.

3. Pruebas de Volumen

   Se realizan estas pruebas para verificar el funcionamiento del sistema bajo altas cargas de datos, realizando para ello simulaciones de cargas de trabajo esperadas.

4. Pruebas de Disponibilidad de Datos

   Con este tipo de pruebas se pretende verificar la capacidad del sistema de recuperarse ante fallos sin que se vean dañados la integridad de los datos.

4 Solución

El objetivo de este apartado consiste en presentar la solución propuesta al problema expuesto en el apartado 1.1 Definición del problema. Este apartado está dividido en varios sub-apartados. El primer apartado está orientado a exponer la descripción de la solución propuesta. También se expondrá de qué forma se abordan los objetivos principales del trabajo presentados en el apartado 1.2 Objetivos. El segundo apartado se centra en el proceso de desarrollo, detallando cada una de las fases realizadas para la creación de la aplicación: fase de análisis, fase de diseño, pruebas e implementación.

4.1 Descripción de la solución

La solución adoptada es una aplicación cliente-servidor en la que el usuario pueda realizar análisis predictivo descriptivo a partir de los datos recogidos en la aplicación de comercio electrónico Magento. En concreto, la solución consiste en una aplicación web, es decir, una herramienta que los usuarios pueden utilizar accediendo a un servidor web a través de Internet mediante un navegador.

A continuación se indica cómo se han cubierto los objetivos expuestos en el apartado objetivos:

> La aplicación debe **permitir realizar un pre procesado de los datos** con el fin de personalizar los reportes obtenidos. Esto se cubre de dos formas distintas. Primero por medio de un apartado de pre-preprocesado por medio de una lista seleccionable de atributos del conjunto a analizar. En segundo lugar, la aplicación permite seleccionar una serie de características propias dependiendo del modelo que se vaya a ejecutar sobre el conjunto de datos; como se ha especificado en apartados anteriores, se va a realizar análisis predictivo descriptivo, es decir, que se realizará el análisis a través de modelos de clasificación y agrupamiento. En la siguiente ilustración se muestran las opciones de configuración y pre-procesado de los conjuntos de datos:

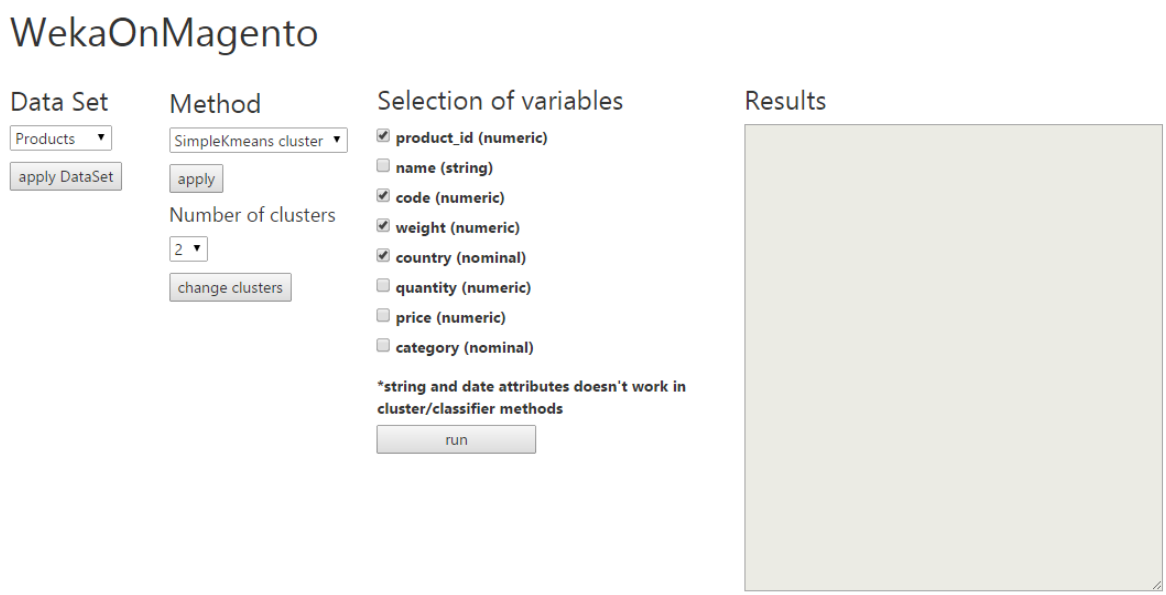


Figura 17: Interfaz de la solución

La aplicación deberá **incluir métodos de análisis predictivo** para conseguir reportes complejos útiles para los gestores de la plataforma. Esto se cubre por medio del soporte de métodos de clasificación agrupamiento de la aplicación web. Ambas técnicas son aplicables a los conjuntos de datos contenidos en la aplicación. En la siguiente ilustración se muestran las opciones de modelado de los conjuntos de datos:

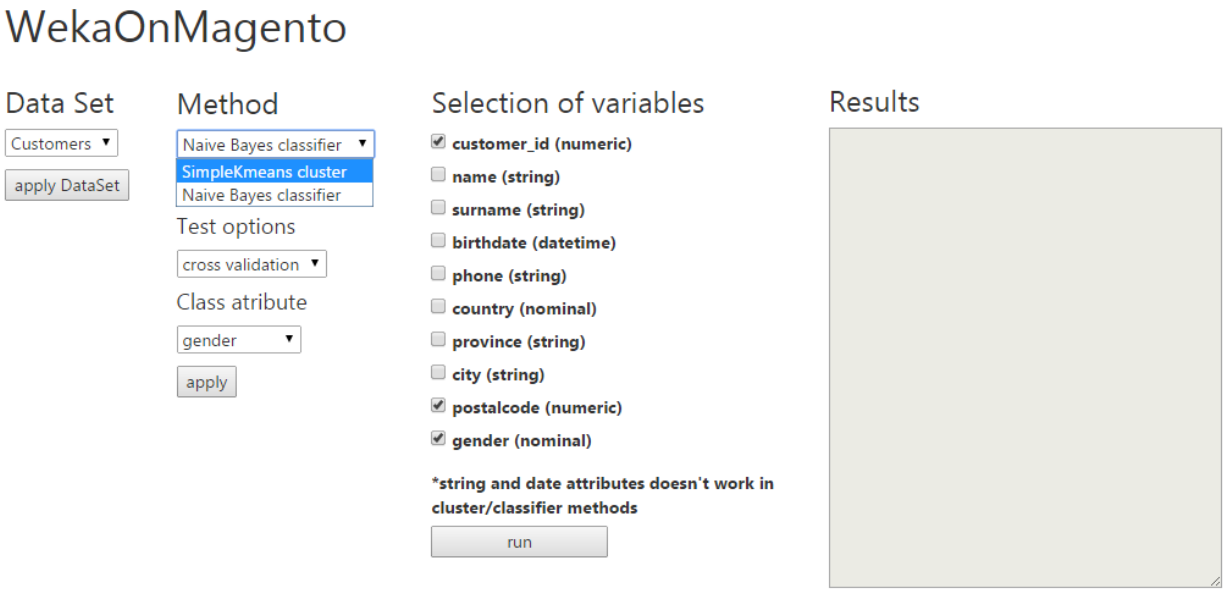


Figura 18: Interfaz de la solución: Selección de método

4.2 El proceso de desarrollo

En este apartado se detalla el proceso de desarrollo que se ha llevado a cabo para realizar el proyecto, detallando el modelo del proceso aplicado para elaborar la solución y las fases de las que consta la misma. El desarrollo consta de cuatro fases fundamentales: la **fase de análisis**, donde se detallan los requisitos del sistema; la **fase de diseño**, donde se identifican los componentes del sistema y sus características; la **fase de implementación**, donde se expone la arquitectura del sistema y su organización; y la **fase de pruebas** donde se detallan las pruebas realizadas sobre la aplicación.

Modelo del proceso

El ciclo de vida de un proyecto software se definen como un marco de referencia en el que se incluyen los procesos, actividades y tareas que forman parte del proceso de desarrollo, mantenimiento y despliegue de un proyecto software. El modelo de ciclo de vida lo componen diferentes fases expresadas en la introducción de este apartado. El modelo de proceso empleado para elaborar la solución ha sido el **modelo en espiral** [ESP], perteneciente a los modelos de desarrollo evolutivo. Este modelo es muy similar al modelo en cascada (que se producen en el que se realiza un desarrollo vertical y las fases se van apoyando en fases anteriores) pero este método tiene la particularidad de que enfatiza en la naturaleza iterativa del proceso de diseño, es decir, utiliza el modelo en cascada para cada etapa. Eso introduce un ciclo de prototipo iterativo, en la que cada iteración, las nuevas expresiones obtenidas a partir de otras son examinadas para cuantificar su proximidad al objetivo deseado. A continuación se muestra un diagrama en el que expresa el ciclo de vida de una aplicación según este modelo:

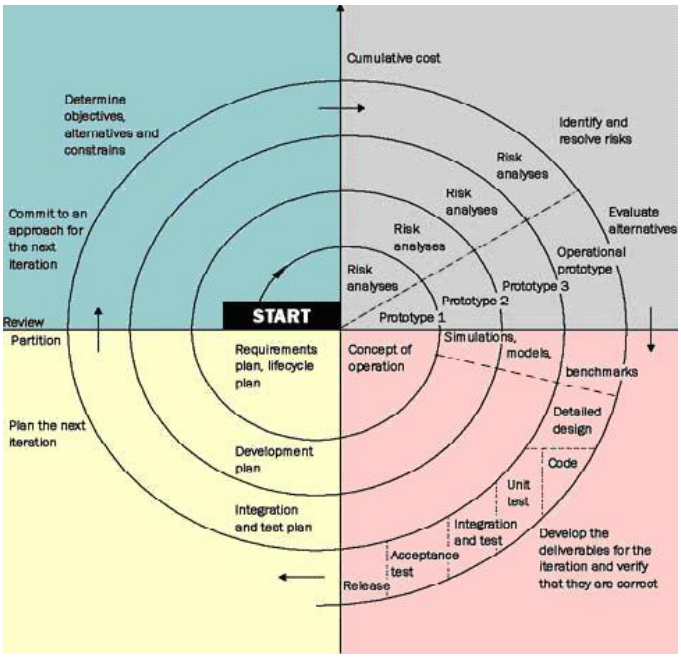


Figura 19: Ciclo de vida del modelo en espiral

Análisis

La fase análisis está dirigida a estudiar las características del problema con el propósito desarrollar una solución que lo resuelva. La fase de análisis simboliza la representación formal de la información, mediante la definición por medio de los requisitos, que servirá de guía para elaborar la solución. Habitualmente, la extracción de requisitos por medio de interacción con el cliente es difícil, debido a que muchas veces es complicado expresar cuáles son las necesidades a cubrir por el sistema. Para definir este listado de requisitos, se utilizará la plantilla de Volere [VOL]. Dicha plantilla ha sido adaptada según las necesidades del sistema es siendo estos, los campos relevantes para el desarrollo de este proyecto:

- **Identificador de requisito (RX-NN):** descriptor que caracteriza unívocamente a un requisito. **R** representa que es un requisito, **X** representa el tipo de requisito (de carácter funcional o no funcional) y **NN** representa el número del requisito.
- **Nombre:** descriptor inicial del requisito.
- **Fuente:** origen del requisito.
- **Descripción:** descripción del requisito.
- **Claridad:** mide la ambigüedad de cada uno de los requisitos. Los valores pueden ser alta, media y baja; siendo de claridad alta un requisito con alto nivel de ambigüedad, y baja, un requisito con bajo nivel de ambigüedad.
- **Prioridad:** define el nivel de importancia del requisito. La prioridad puede ser alta, media o baja, dependiendo del grado de relación que tenga dicho requisito respecto a

los objetivos del proyecto. Es decir, los requisitos tendrán una prioridad alta si están fuertemente ligados a los objetivos del proyecto, y tendrán una prioridad baja si tienen una leve relación con los objetivos.

A continuación se muestra la plantilla empleada para realizar la especificación de requisitos:

RX – XX	
Nombre	
Fuente	
Prioridad	
Claridad	
Descripción	

Tabla 14: Adaptación de la plantilla de requisitos de Volere

Definición de requisitos

En el presente apartado, se procederá a enumerar todos los requisitos del sistema clasificados entre funcionales y no funcionales. Identificaremos los requisitos según el identificador de requisito definido en el apartado anterior

Requisitos funcionales

Los requisitos funcionales representan las declaraciones de los servicios que debe proporcionar el sistema, es decir, representan las capacidades a cubrir por el sistema.

RF-01: se le permitirá al usuario seleccionar el conjunto de datos.

RF-02: se le permitirá al usuario seleccionar el método de aplicación.

RF-03: se le permitirá al usuario seleccionar los atributos del conjunto de datos con los que desea operar

RF-04: la aplicación permitirá la identificación de usuarios.

RF-05: se le permitirá al usuario seleccionar el método de test cuando utilice un método de clasificación.

RF-06: se le permitirá al usuario seleccionar el atributo de clase cuando utilice un método de clasificación.

RF-07: se le permitirá al usuario seleccionar el número de agrupaciones cuando utilice un método de agrupamiento.

RF-08: la aplicación mostrará los resultados.

RF-09: se mostrará el árbol cuando se ejecute una tarea de clasificación.

RF-10: se mostrarán los grupos a los que pertenece cada instancia del conjunto de datos.

RF-11: se permitirá registrarse en la aplicación.

Requisitos no funcionales

Los requisitos funcionales representan las declaraciones de forma respecto a los servicios que debe proporcionar el sistema, representan servicios y funciones ofrecidas por el sistema

RNF-01: la totalidad de la interfaz se mostrará en inglés.

RNF-02: el entorno debe de poder desplegarse en cualquier navegador web.

RNF-03: se utilizará una interfaz sencilla.

RNF-04: el interfaz contará con cuatro columnas: Data Set, Method Selection of Attributes y Results.

RNF-05: la selección de los conjuntos de datos se realizará por medio de un menú desplegable.

RNF-06: el entorno será visualizado un en un monitor de mínimo 15 pulgadas.

RNF-07: se le permitirá al usuario seleccionar un solo conjunto de datos con cada ejecución del modelo.

RNF-08: se le permitirá al usuario seleccionar un solo método con el que desea operar.

RNF-09: los atributos del conjunto de datos se seleccionaran de forma múltiple.

RNF-10: se le permitirá al usuario seleccionar el método de test entre cross validation y training test.

RNF-11: se le permitirá al usuario seleccionar el atributo de clase entre los atributos contenidos en el conjunto de datos seleccionado.

RNF-12: se le permitirá al usuario seleccionar un número de clusters contenido entre 2 y 5.

RNF-13: debe existir coordinación entre los datos del modelo de Magento y los conjuntos de datos utilizados por la aplicación.

RNF-14: el sistema contará con los conjuntos de datos de productos y clientes.

RNF-15: debe existir coordinación entre las credenciales introducidas y los usuarios almacenados en la base datos.

RNF-16: el conjunto de datos de *Products* contará con las siguientes variables: *product_id, name, code, weirht, country, quantity, price y category.*

RNF-17: el conjunto de datos *Customers* contará con las siguientes variables: *customer_id, name, surname, birthdate, pone, country, province, city, postalcode y gender.*

Casos de Uso

En esta fase se presentan los casos de uso obtenidos a partir de la definición de requisitos en las páginas anteriores. De manera general, un caso de uso no es más que una descripción de los pasos que el usuario debe realizar para llevar a cabo un proceso. En este caso concreto, sirven para definir los pasos que tiene que llevar a cabo un usuario para cumplir la funcionalidad descrita en los requisitos.

A continuación se mostrará el Diagrama de Casos de Uso en el que contamos con 2 actores: usuario registrado y usuario no registrado.

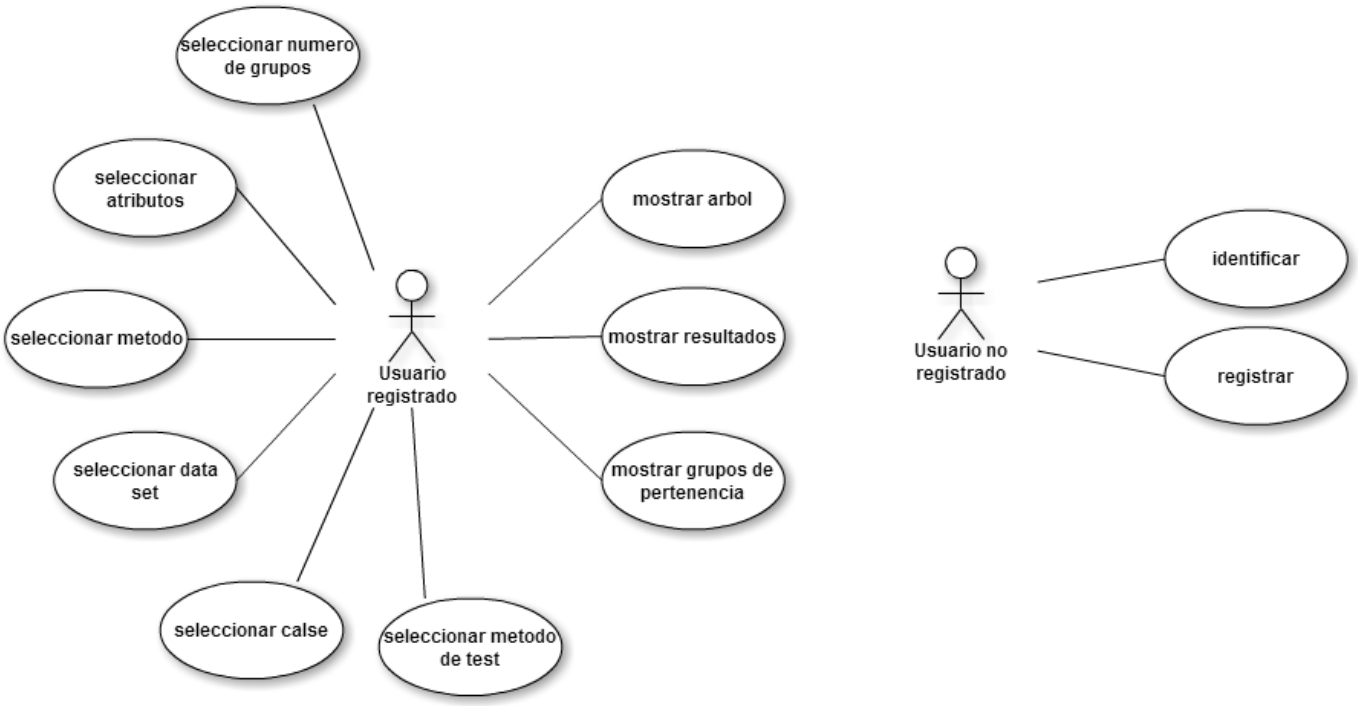


Figura 20: Diagrama de Casos de Uso

A continuación se presentan los campos relevantes para la correcta identificación de los casos de uso:

- **Identificador de requisito (CU-NN):** escritor que caracteriza unívoca mente a un caso de uso. **CU** representa que es un Caso de Uso y **NN** representa el número del Caso de Uso.
- **Nombre:** descriptor inicial del Caso de Uso.
- **Actores:** Entidad externa al sistema que demanda la funcionalidad dada por el caso de uso**.**
- **Escenario:** Explica en qué contexto se aplica el caso de uso.
- **Precondiciones:** Condiciones que se deben cumplir para que el curso de eventos pueda llevarse a cabo.
- **Curso básico:** Especifica la interacción entre los actores y el sistema.

A continuación, se muestran las matrices de trazabilidad que relacionan cada uno de los requisitos presentados en el apartado definición de requisitos con los casos de uso que se han presentado en el presente apartado. Se presentarán dos matrices en función de la naturaleza de los requisitos que se trazan con los casos de uso.

	CU-01	CU-02	CU-03	CU-04	CU-05	CU-06	CU-07	CU-08	CU-09	CU-10	CU-11
RF-01	X		X					X	X	X	
RF-02		X			X	X	X	X	X	X	
RF-03			X					X	X	X	
RF-04				X							X
RF-05					X			X	X		
RF-06						X		X	X		
RF-07							X	X		X	
RF-08								X			
RF-09								X	X		
RF-10										X	
RF-11				X							X

Tabla 15: Matriz de trazabilidad: Requisitos funcionales vs. Casos de Uso

	CU-01	CU-02	CU-03	CU-04	CU-05	CU-06	CU-07	CU-08	CU-09	CU-10	CU-11
RNF-01											
RNF-02											
RNF-03											
RNF-04											
RNF-05	X										
RNF-06											
RNF-07	X										
RNF-08		X									
RNF-09			X								
RNF-10					X						
RNF-11						X					
RNF-12							X				
RNF-13	X										
RNF-14	X										
RNF-15				X							X
RNF-16	X		X								
RNF-17	X		X								

Tabla 16: Matriz de trazabilidad: Requisitos no funcionales vs. Casos de Uso

Diseño

Esta fase permite establecer la solución a elaborar, identificando nuestras garantías esenciales de la misma como las características detalladas. En esta fase se detalla el seguimiento del modelo del proceso de desarrollo expuesto anteriormente, el modelo en espiral, mediante el desarrollo en cascada de cada uno de las fases. Se desarrolla un pequeño prototipo y se evalúan las alternativas y los riesgos.

Diseño de sistema

A continuación se detallarán cada una de las tecnologías utilizadas para el desarrollo del proyecto separado por 2 categorías: plataforma de aplicación web (que se evalúan en profundidad en el apartado Tecnología para desarrollo de aplicación cliente servidor y herramienta de generación de modelos (que se evalúa en profundidad en el apartado Tecnología para generación de modelos)

Aplicación cliente-servidor

El entorno desarrollado se trata de un modelo de aplicación distribuida en la que las tareas se reparten entre servidor y cliente. El servidor es el encargado de proveer los recursos y servicios que son demandados por el cliente.

La aplicación web, ha sido desarrollada en Ruby (versión 1.9.3) sobre el entorno de Rails (versión 4.0.2). El motivo de utilizar la plataforma de Ruby on Rails ha sido principalmente por dos razones: la posibilidad de tratar el modelado de datos de manera independiente al resto de componentes y permitir escalabilidad en cuanto a la construcción del modelo a través de la generación de migraciones para modificar la estructura de los mismos tal y como se describe en el apartado Tecnología de desarrollo de aplicación cliente-servidor.

Herramienta de generación de modelos

Bajo el entorno de desarrollo escrito en el apartado anterior, se ha integrado la herramienta de generación de modelos Weka (versión 3.6.11). Esta herramienta se ha integrado a través del módulo de Rails rjb (*ruby-java-bridge*). Este módulo o gema permite establecer un middleware entre el código Ruby y las llamadas a la herramienta Java

Diseño detallado

En este apartado se describe el diseño de las capas de MODELO e INTERFAZ del sistema. Se realizará un estudio detallado de cada una de las capas con el fin de plasmar los componentes de diseño utilizados en el proyecto.

Diseño del modelo de datos

En el diseño del modelo de datos, se han generado tres modelos de datos: Products y Customers haciendo referencia a los conjuntos de datos siguiendo las exigencias reflejadas en los requisitos no funcionales RNF-16 y RNF-17 respectivamente; y User para la gestión de usuarios.
A continuación se mostrarán los atributos referentes a cada uno de los modelos de datos:

Product	
Atributo	Tipo
product_id	entero
name	cadena
code	cadena
weight	real
country	cadena
quantity	entero
price	real
category	cadena
timestamps	-

Tabla 17: Atributos del modelo Product

El campo timestamps guarda los atributos temporales de la creación del modelo. Esto resulta muy útil a la hora de establecer un marco temporal de los datos y poder hacer seguimientos de los mismos cuando se han realizado Migraciones [MIG]

Customer	
Atributo	Tipo
customer_id	entero
name	cadena
surname	cadena
birthdate	fecha
email	cadena
phone	cadena
country	cadena
province	cadena
city	cadena
postalcode	entero
gender	entero
timestamps	-

Tabla 18: Atributos del modelo Customer

Diseño de los Interfaces

En este apartado, se especificarán cada una de las fases de cada una de los interfaces utilizados en el sistema. Contamos con tres interfaces principales involucradas en el proyecto: interfaz principal (main) y dos interfaces referentes a la identificación y registro usuario: registrar (Sign up) e identificar (Sign in). La interfaz de gestión de usuarios representa el interfaz de identificación y el de registro; y la vista principal hace referencia al interfaz en el que se realiza el análisis predictivo.

Las interfaces de gestión de usuarios sean mantenido fieles al diseño especificado en este apartado:

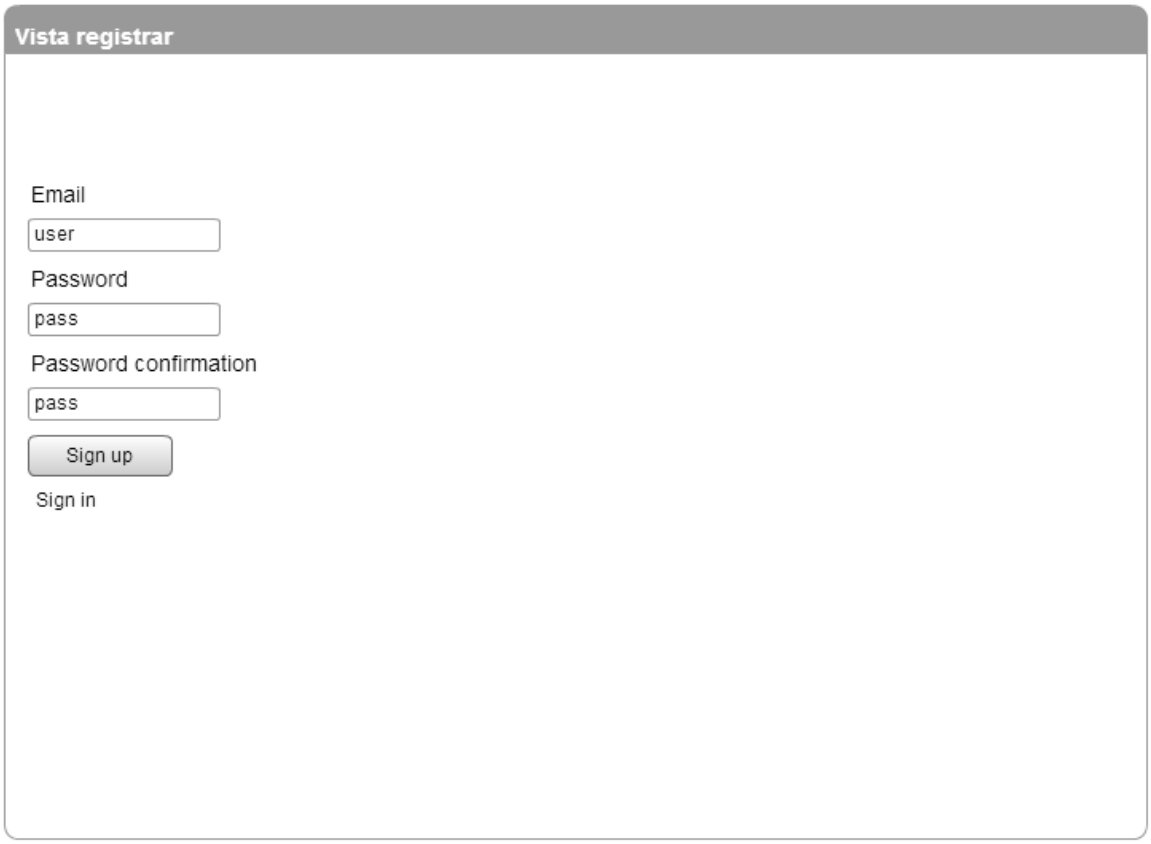


Figura 21: Diseño de Interfaz registrar

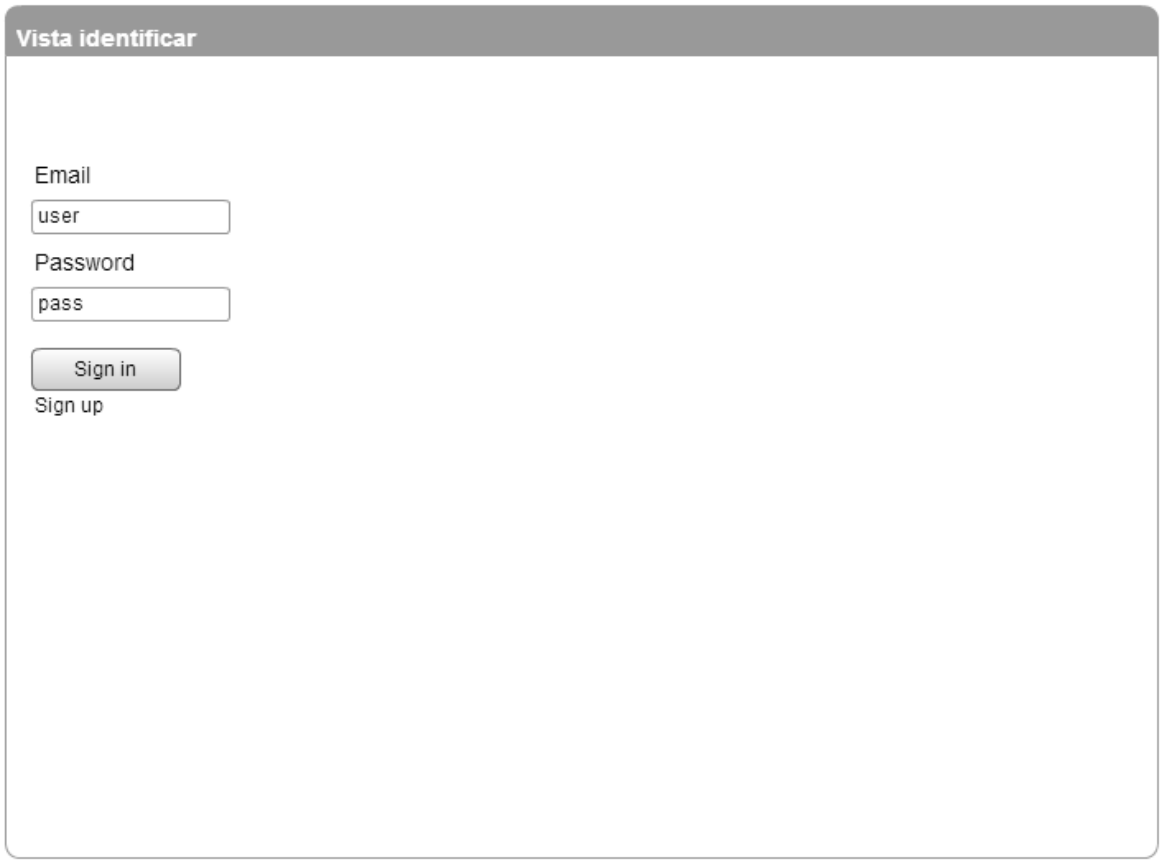


Figura 22: Diseño de interfaz identificar

En el caso del interfaz principal, se han tenido que realizar diferentes diseños en función de la fase de implementación en la que se encontrara el proyecto. A continuación se mostrarán los diseños de cada una de las vistas.

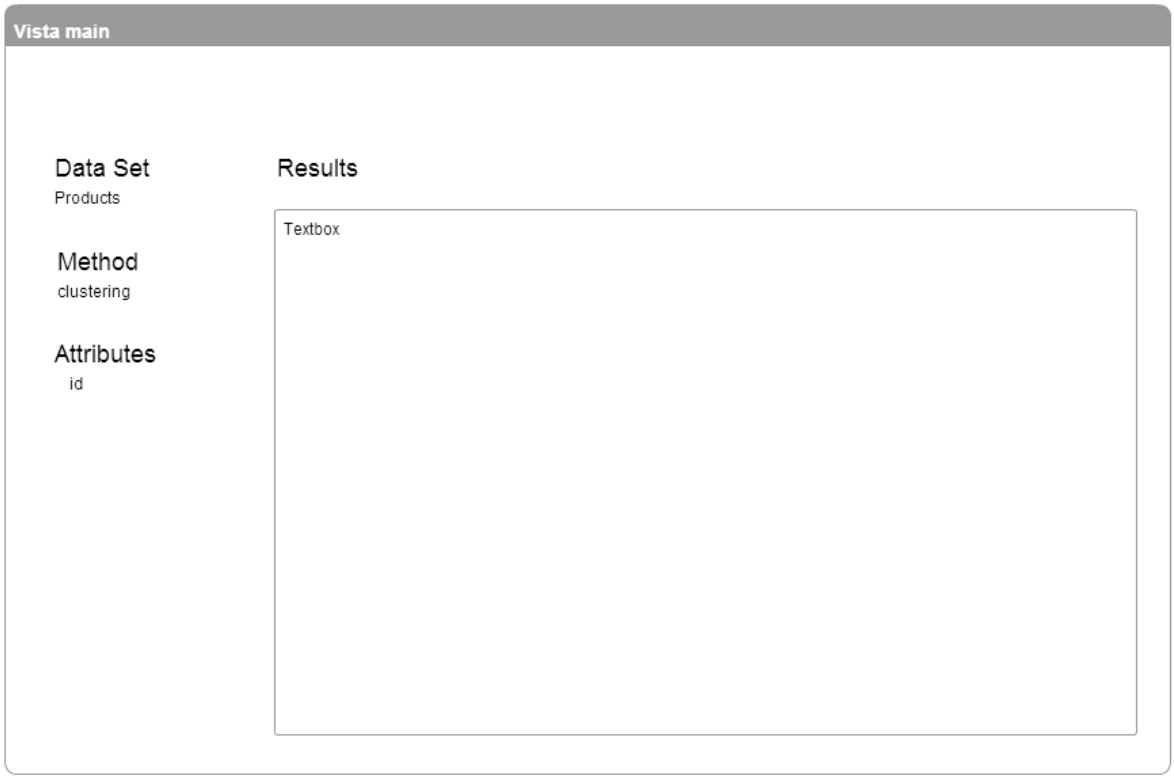


Figura 23: Diseño de Interfaz main Fase 1

La Fase 1 del interfaz *main* corresponde con la interfaz utilizada en la fase de implementación 4, en la que se cierra una versión funcional de la aplicación. Se utiliza el dataset *Products*, solo se tiene en cuenta el id para generar el modelo y se utiliza una tarea de agrupamiento.

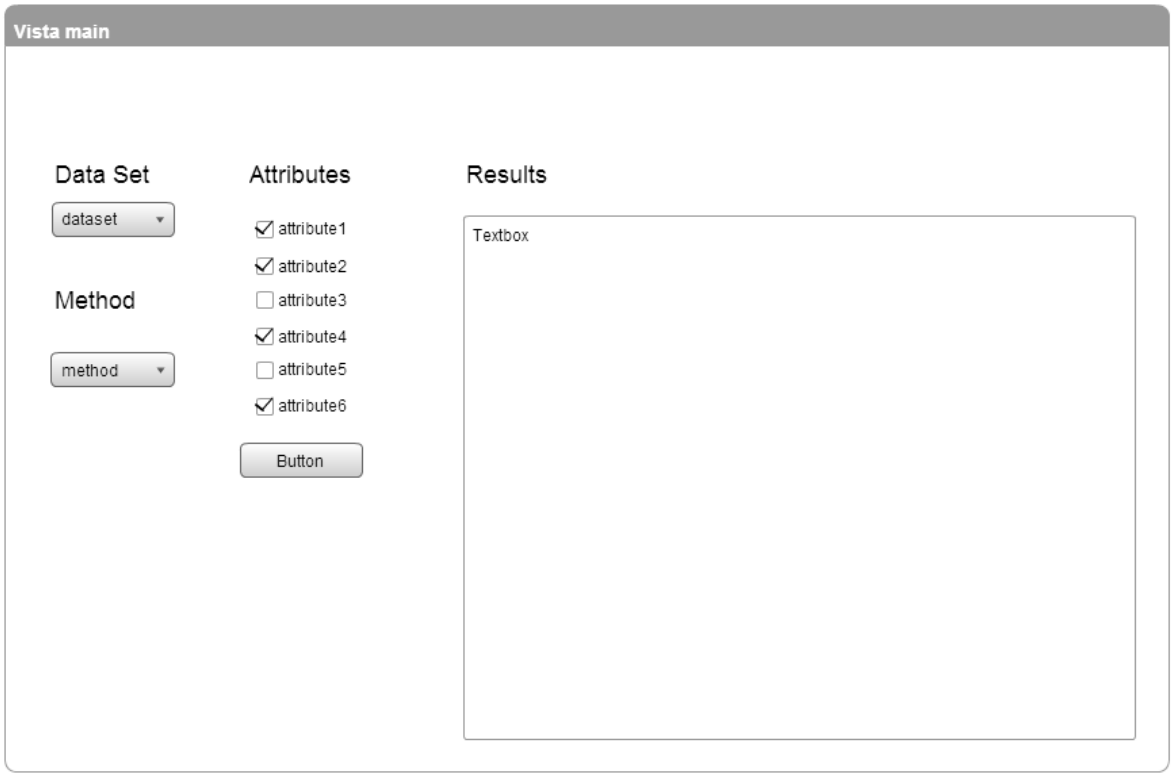


Figura 24: Diseño de Interfaz main Fase 2

La Fase 2 del interfaz *main* también corresponde con la interfaz utilizada en la fase de implementación 4, solo que en este caso se incluye la funcionalidad de seleccionar los atributos y se incluye el modelo de clasificación a los métodos.

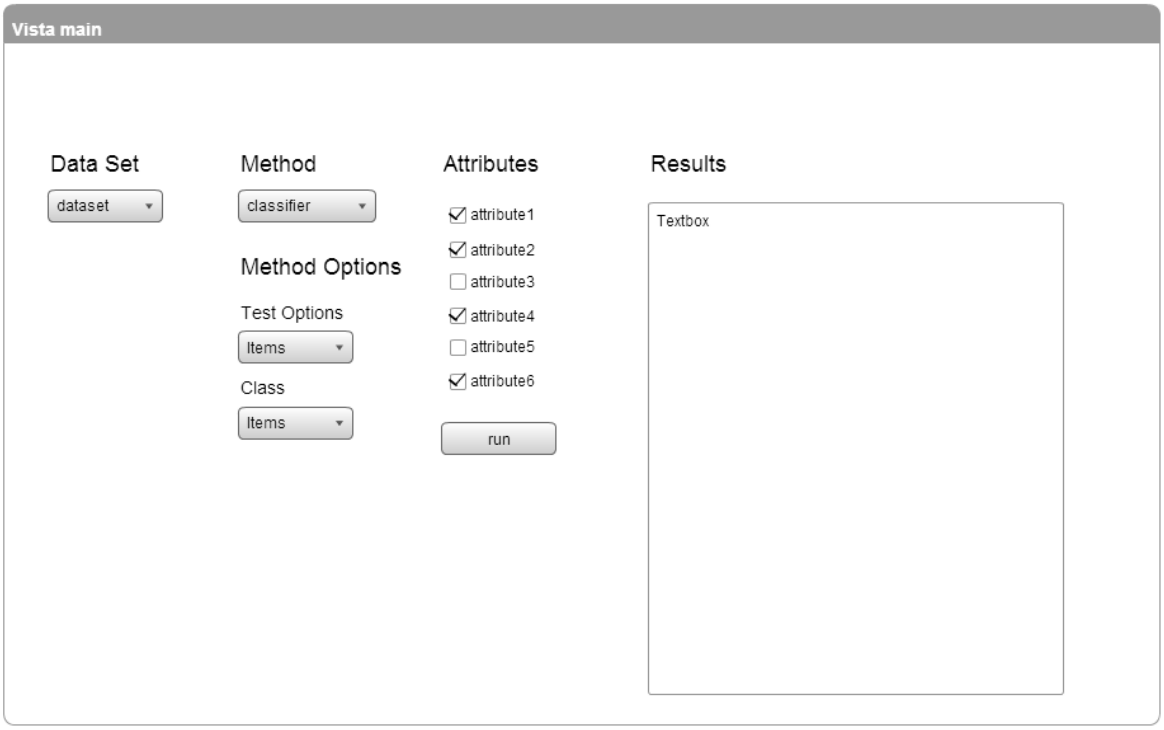


Figura 25: Diseño de interfaz main fase 3

La Fase 3 del interfaz main corresponde con la interfaz utilizada en la fase de implementación 5. Esta sería la interfaz final en la que se añaden las opciones de modificación de los modelos.

Implementación

Esta fase está destinada a conocer las características del proyecto desarrollado, identificando la arquitectura del proyecto. La primera parte de este apartado se va a dedicar a explicar la arquitectura MVC (Modelo-Vista-Controlador) que se ha aplicado en este proyecto así como la justificación de sus componentes. La segunda parte está destinada a identificar las características de la arquitectura en el proyecto desarrollado.

Arquitectura MVC

El patrón de arquitectura MVC [MVC] define la organización independiente del **Modelo** (objetos de negocio), la **Vista** (interfaz de interacción con el usuario u otro sistema) y el **Controlador** (que define el flujo de trabajo de la aplicación). De esta manera, podemos subdividir el sistema en tres capas, consiguiendo tratar, de manera independiente: los datos y la lógica de negocio, la interfaz y el módulo de gestión de eventos y comunicaciones. De esta forma se consigue facilitar la tarea de desarrollar aplicaciones y su posterior mantenimiento.

A continuación se explicarán los tres componentes principales de este modelo:

- **Modelo:** constituye la representación de la información con la que opera el sistema y gestiona todos los accesos a dicha información implementando también los privilegios de acceso descritas en la lógica de negocio. El Modelo envía a la Vista la información que solicita para mostrar. Las peticiones de acceso o manipulación de información llegan al Modelo a través del Controlador.

- **Vista:** presenta el Modelo en un formato adecuado para interactuar con él, es decir la interfaz de usuario.

- **Controlador:** es el encargado de responder a los eventos (normalmente interacciones del usuario con el interfaz) e invocar peticiones al modelo cuando se realiza alguna solicitud de información. El controlador funciona como intermediario o *middleware* entre la Vista y el Modelo.

Estas son las relaciones entre una vista, un modelo y un controlador:

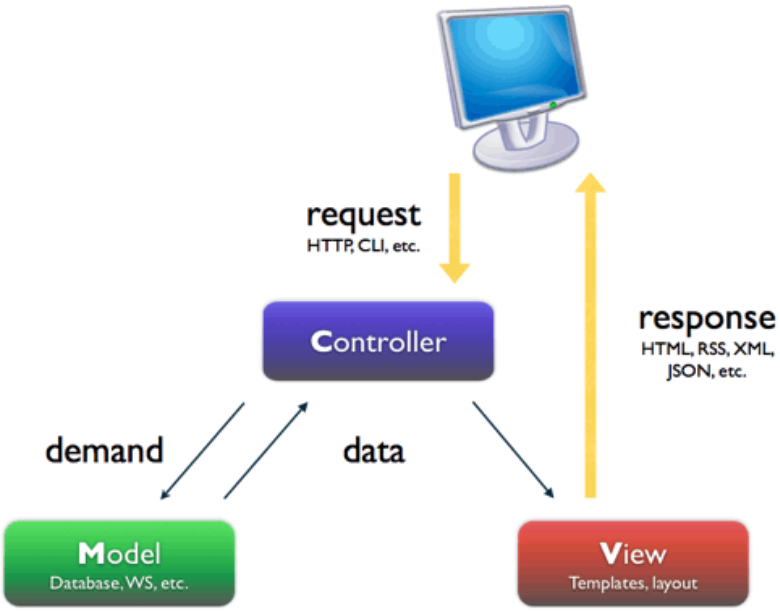


Figura 26: Esquema de arquitectura MVC

Esta arquitectura presenta grandes de beneficios:

- Permite facilitar la conversión del código. El código está muy encapsulado y actúa de manera casi independiente. Esto permite que su aprendizaje sea más rápido y reutilizar el código para otras aplicaciones.

- Vistas actualizadas: el programador no debe ocuparse de solicitar que las vistas actualicen, ya que este proceso se realiza automáticamente por el modelo de la aplicación.

- Cualquier modificación sujeta al dominio implica una modificación sobre el modelo y las interfaces del mismo, es decir, no es necesario modificar los mecanismos de comunicación y actualización de modelos contemplados en el controlador.

- Las modificaciones de la lista no afectará al modelo, simplemente se modifica la representación de información, no su tratamiento.

Las aplicaciones desarrolladas bajo esta arquitectura presentan una extensa flexibilidad y una mantenibilidad únicas comparadas con aplicaciones basadas en otros patrones.

Modelo

Para la realización del modelo de datos, ha sido necesario mapear los datos recogidos en el modelo de datos de la plataforma Magento. Como ya se explicó en el apartado Modelo de datos EAV, el modelo de datos de Magento es un modelo poco convencional por lo que será necesario adaptar dichos datos al modelo relacional tradicional con el que trabaja Ruby on Rails.
La aplicación desarrollada necesita tener acceso a dos bases de datos distintas: la BBDD de Magento de tipo MySQL y la BBDD de la aplicación de tipo SQLite. A continuación se presenta el diagrama de componentes del modelo de datos de la aplicación siguiendo la arquitectura MVC.

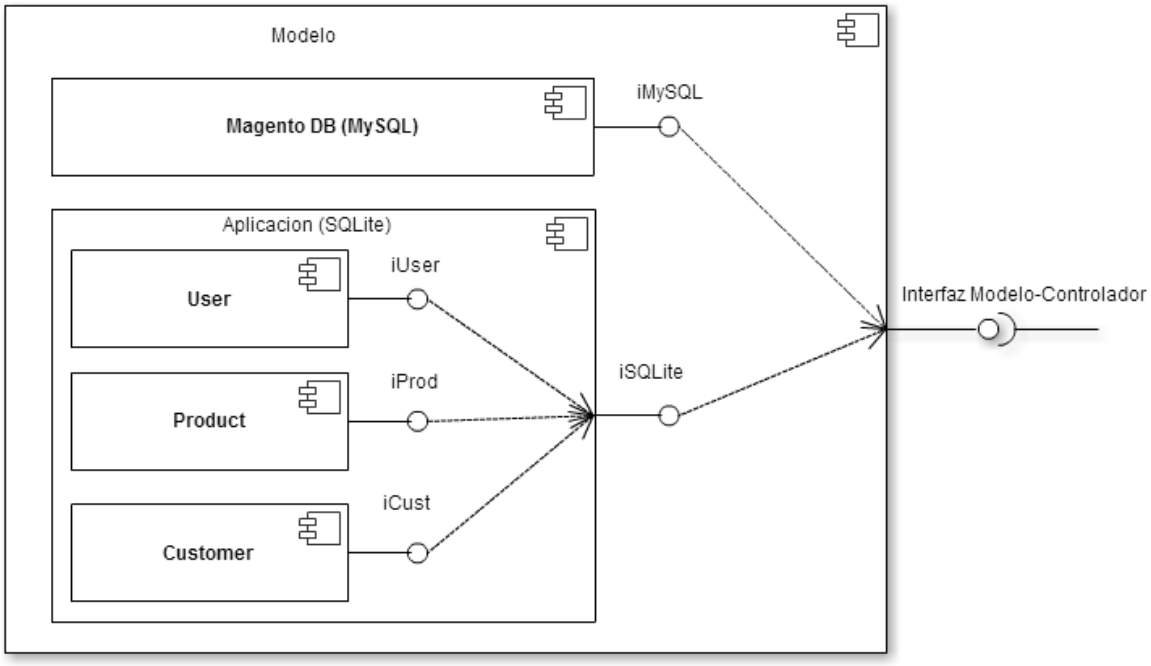


Figura 27: Diagrama de componentes: Modelo

Si analizamos la estructura de carpetas de un proyecto en Rails, podemos observar que toda la información referente a los modelos se encuentran en dos carpetas principales: app/models y db/migrate.

En app/models, se encuentran todos los modelos generados en la aplicación en Rails. Dentro de estos ficheros de modelo se especifican las referencias a la BBDD y se definen las características de los datos, como por ejemplo que campos son imprescindibles para almacenar una instancia de ese modelo.

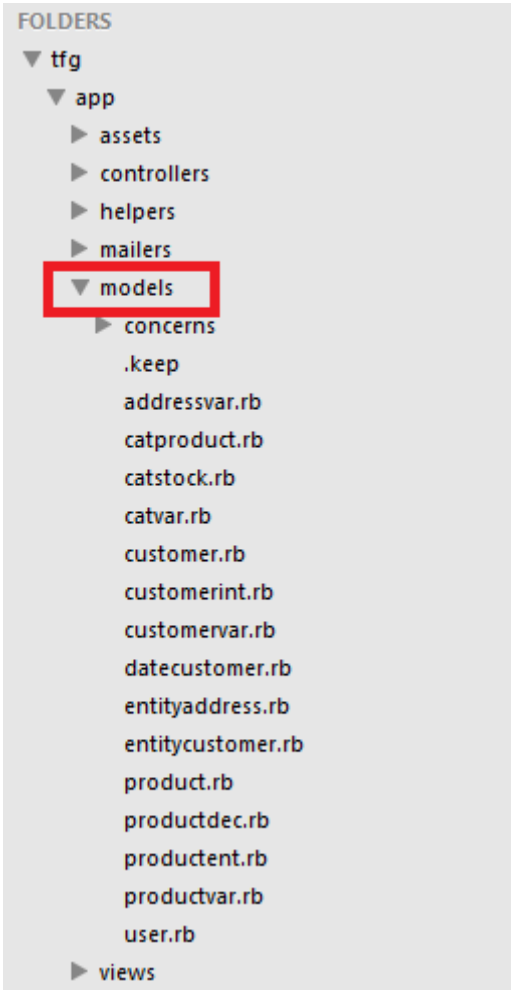


Figura 28: Estructura de un proyecto Rails: modelos

En db/migrate, se encuentran todas las migraciones que se han generado en la aplicación en Rails. Las migraciones constituyen uno de los elementos más potentes del desarrollo en Rails debido a que almacena el estado de un modelo en un momento concreto (van nombrados la fecha de la migración y por el modelo). Una vez almacenada una migración, se pueden efectuar nuevas migraciones modificando dicho modelo y, lo más importante, **las migraciones permiten revertir los cambios efectuados en los modelos**

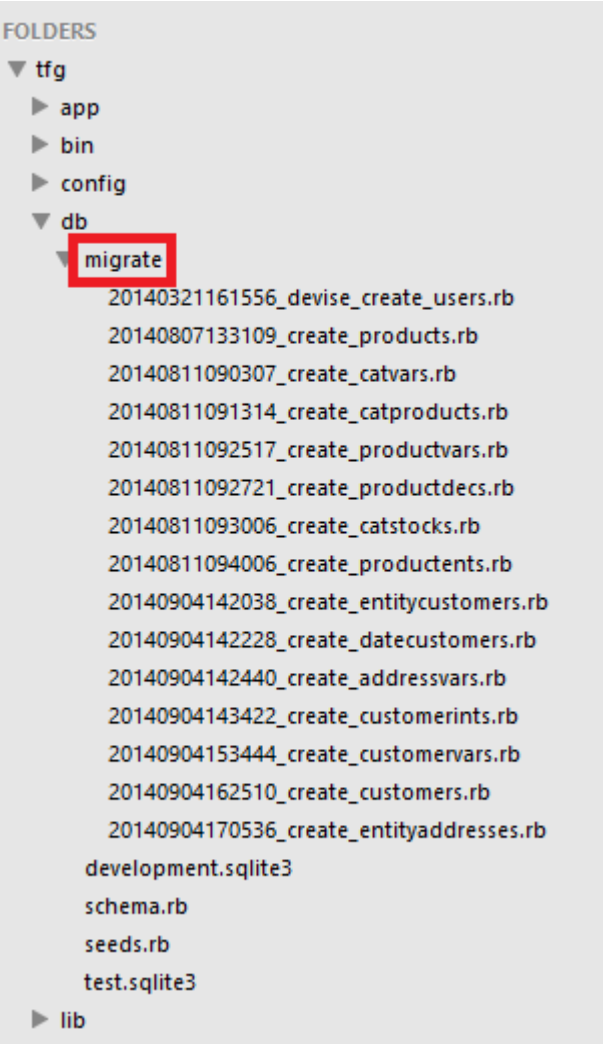


Figura 29: Estructura de un proyecto Rails: migraciones

Siguiendo el paradigma de programación de Rails, ha sido necesario establecer un modelo por cada tabla de la que se ha querido extraer información. Como ya mencionamos en el apartado referente al modelo de datos EAV, para extraer un dato del modelo es necesario relacionar numerosas tablas; pues bien, ha sido necesario mapear en un modelo de Rails cada tabla necesaria para obtener los datos.

Primero se ha generado los modelos Product y Customer en Rails con los atributos mencionados en el apartado anterior, Diseño del modelo de datos. Para adaptar los datos de la base de datos de Magento a los modelos en Rails, se han generado numerosos modelos auxiliares en función del conjunto de datos sobre el que se van a obtener los datos. A continuación se mostraran los modelos auxiliares generados en función del conjunto de datos:

Modelos auxiliares para el mapeo de datos desde Magento al modelo **Products** de Rails:

- Modelo auxiliar **Productvar**: mapea la tabla **catalog_product_entity_varchar**.
- Modelo auxiliar **Catstock**: mapea la tabla **cataloginventory_stock_item**.
- Modelo auxiliar **Productdec**: mapea la tabla **catalog_product_entity_decimal**.
- Modelo auxiliar **Catproduct**: mapea la tabla **catalog_category_product**.
- Modelo auxiliar **Catvar**: mapea la tabla **catalog_category_entity_varchar**.
- Modelo auxiliar **Productent**: mapea la tabla **catalog_product_entity**.

Modelos auxiliares para el mapeo de datos desde Magento al modelo **Customers** de Rails:

- Modelo auxiliar **Addressvar**: mapea la tabla **customer_address_entity_varchar**.
- Modelo auxiliar **Customerint**: mapea la tabla **customer_entity_int**.
- Modelo auxiliar **Customervar**: mapea la tabla **customer_entity_varchar**.
- Modelo auxiliar **Datecustomer**: mapea la tabla **customer_entity_datetime**.
- Modelo auxiliar **Entityaddress**: mapea la tabla **customer_address_entity**.
- Modelo auxiliar **Entitycustomer**: mapea la tabla **customer_entity**.

No es necesario generar ningún modelo auxiliar para mapear los datos de usuario ya que se gestiona según el modelo relacional tradicional en una base de datos SQLite.

Ningún modelo auxiliar ha sido incluido en el modelo del diagrama de componentes debido a que han sido creados exclusivamente para trabajar con los datos según las convenciones de Rails. Dichos modelos actúan como estructuras auxiliares y no presentan ningún cambio sobre ninguna de las dos bases de datos con las que trabaja la aplicación.

Vista

Como ya se mostró en el diseño detallado, la aplicación cuenta con 3 interfaces; dos de ellos son manejados por el controlador de login (login_controller) y la vista principal (main) es controlada por el controlador principal (main_controller). A continuación se muestra el diagrama de componentes referentes a la vista:

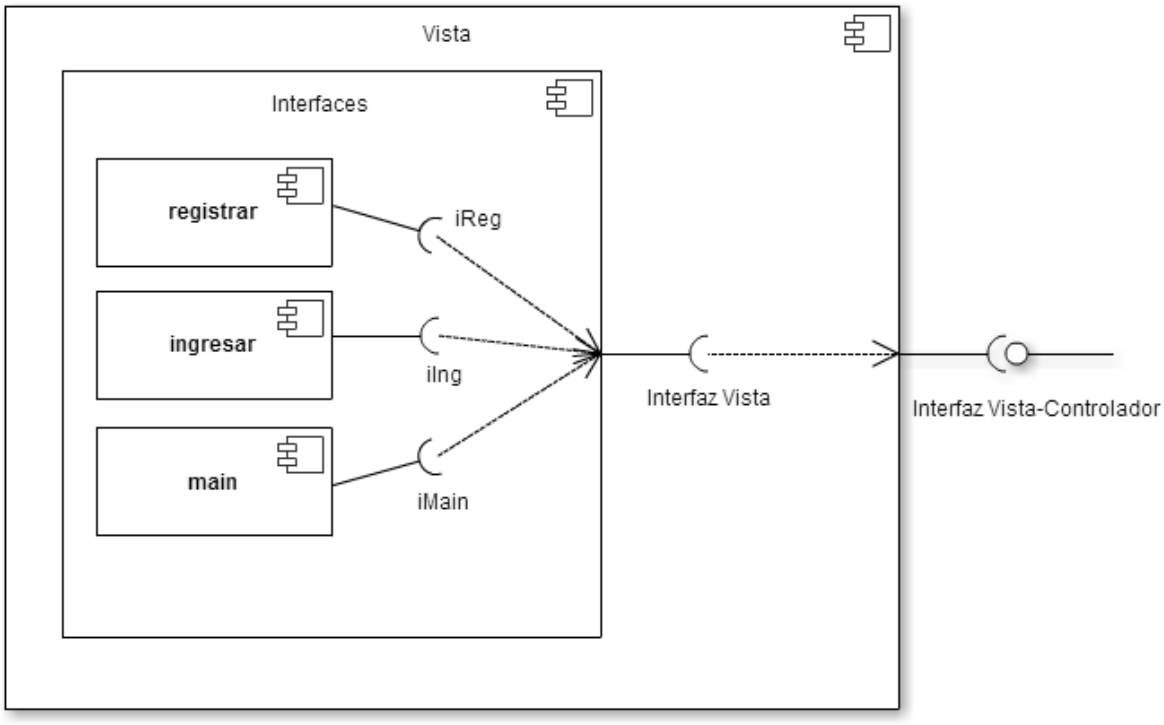


Figura 30: Diagrama de components: Vista

Por convención, Rails relaciona como controlador de una vista, al controlador cuyo nombre sea el "nombre de la vista"*_controller*; es decir, el controlador de la vista *main*, será *main_controller*.
Las vistas las podemos encontrar en un proyecto Rails en la carpeta *views*. Aquí podemos encontrar una carpeta para cada vista donde se definen los interfaces.

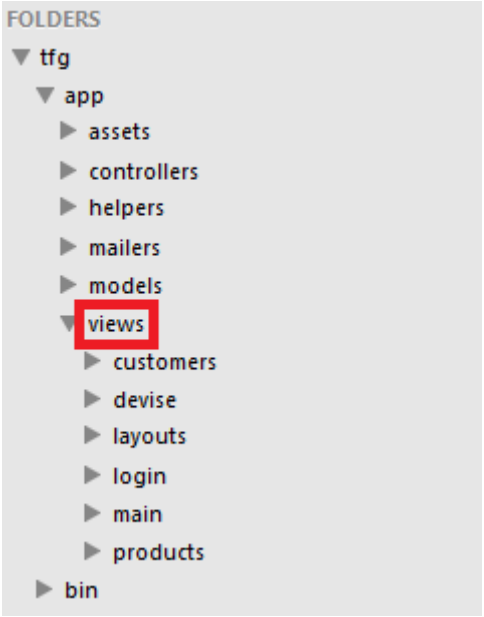


Figura 31: Estructura de un proyecto Rails: vistas

Controlador

En este apartado se especifican los controladores de la aplicación. Los controladores son los componentes encargados de agregar la lógica de la aplicación. Esto engloba los eventos, las peticiones y todo lo que tenga que ver con interacción en la aplicación. A continuación se mostrara el diagrama de componentes de los controladores de la aplicación.

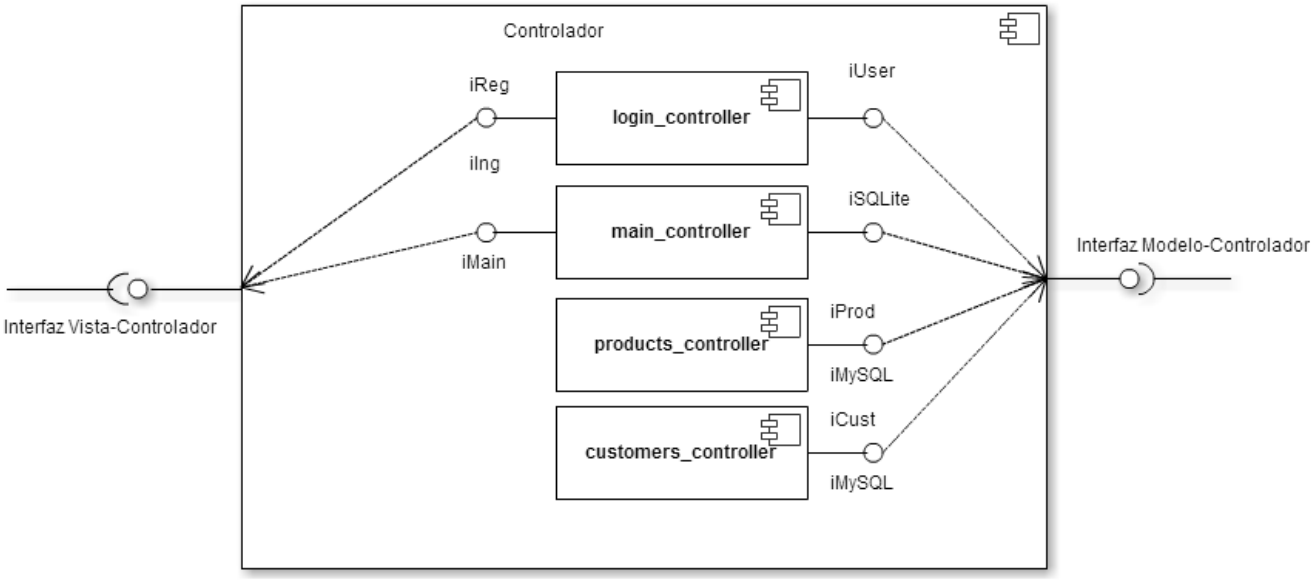


Figura 32: Diagrama de componentes: Controlador

Como se en el diagrama de componentes anterior, existen dos tipos de controladores según su naturaleza: controladores relacionados tanto con las vistas como con los modelos y controladores únicamente relacionados con los modelos. El caso de estos dos controladores que se relacionan con los modelos, hace referencia a la parte de mapeado de datos. Tanto el controlador *products_controller* como el controlador *customer_controller*, son los encargados de extraer los datos contenidos en los modelos auxiliares para generar los modelos que representan los conjuntos de datos con los que trabajará nuestra aplicación.

Una de las peculiaridades de esta aplicación es que necesita conectarse con dos bases de datos distintas de la base de datos de Magento (MySQL) y la base de datos de la aplicación (SQLite). No es algo habitual, que una aplicación trabaje con dos bases de datos diferentes por lo que esta tarea se gestiona de forma un poco peculiar. Rails posee un sistema de comunicación con base de datos llamado *Active Record* [ACR]. Para este caso particular y si nos centramos en los controladores products_controller y customer_controller, es necesario incluir dos métodos en cada uno de los controladores para cambiar la conexión de la base de datos por medio Active Record.

El controlador *main_controller*, es el encargado de agrupar toda la interacción del usuario registrado con la aplicación, es decir, en él se incluye toda la funcionalidad del análisis predictivo. A continuación se enumerarán las funciones que contiene el método *main_controller*:

- **Weka_results:** devuelve los resultados reportados por la tarea análisis.
- **Charge_products:** se encarga de llamar al constructor de products_controller para realizar la carga de los productos almacenados en la base de datos de Magento.
- **Charge_customers:** se encarga de llamar al constructor de customers_controller para realizar la carga de los productos almacenados en la base de datos de Magento.
- **ChangeDataSet:** se encarga de cambiar al conjunto de datos seleccionado por el usuario por medio de la interfaz.
- **ChangeMethod:** se encarga de cambiar el método seleccionado por el usuario por medio de la interfaz.
- **ChangeClusters:** se encarga de cambiar el número de clusters seleccionado por el usuario por medio de la interfaz.
- **ChangeClssOptions:** se encarga de cambiar el atributo de clase seleccionado por el usuario por medio de la interfaz.

- **Main:** vista principal de carga de interfaz.
- **Run:** se encarga de generar el modelo en función de las características introducidas a través del interfaz por el usuario.
- **Kmeans:** se encarga de generar un modelo de agrupamiento con el método k-medias.
- **Classifier:** se encarga de generar un modelo de clasificación con el método Naive Bayes.
- **GenerateArff:** se encarga de generar el fichero donde se almacenan las tuplas recogidas por el modelo, en el formato en el que Weka recoge el conjunto de instancias.
- **Change_rails_database:** se encarga de establecer la conexión con la base de datos SQLite por medio de Active Record.
- **Is_first:** método auxiliar para saber si el valor se encuentra el primero de la lista.
- **Is_middle_or_last:** método auxiliar para saber si el valor se encuentra en algún punto intermedio de la lista o al final.

El controlador login_controller, es el encargado de gestionar todas las tareas de identificación y registro de usuarios.

Los controladores las vistas y los modelos se relacionan directamente en Rails por medio de una convención de nombres. Esta convención sigue la siguiente estructura:

- Modelo: “Nombre”, singular y la primera letra en mayúsculas
- Vista: “nombres”, plural y todo en minúscula
- Controlador: “nombres_controller”, plural, todo en minúscula y añadiendo “_controller”

Si se sigue esta convención en Rails, no es necesario conectar los componentes entre si debido a que se conectarían por defecto. Existen varios problemas con esto, entre otro los plurales irregulares, por lo que Rails también tiene la posibilidad de especificar el nombre de los componentes relacionados

Los controladores los podemos encontrar en un proyecto Rails en la carpeta *controllers*. Aquí podemos encontrar todos los controladores del proyecto.

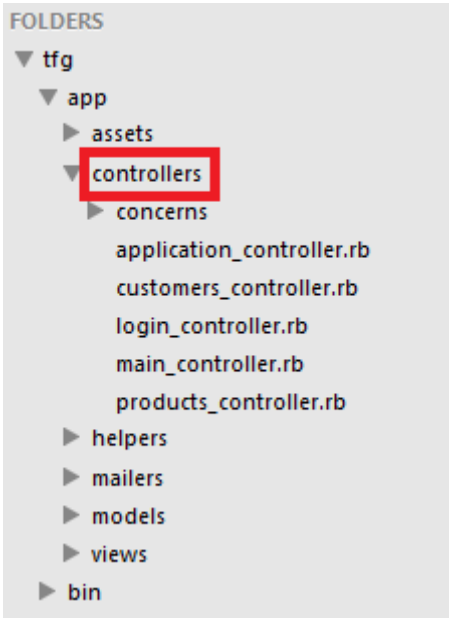


Figura 33Estructura de un proyecto Rails: controladores

El controlador application_controller, es un controlador que viene por defecto en Rails y en él se especifican la funcionalidad común a todos los controladores.

5 Evaluación

Este apartado de evaluación tiene como objetivo demostrar la validez de la solución elaborada. Para que la solución se considere válida, debe resolver los problemas expuestos en la sección 1.1 Problem definition así como satisfacer los objetivos definidos en la sección 1.2 Objetives.

En este apartado se evalúa si el sistema desarrollado es válido a través de una serie de pruebas de validación. En primer lugar se presentará el proceso de evaluación llevado a cabo para validar el sistema. Más adelante se estudiarán diferentes casos de prueba y una vez definido esto, se procederá a extraer conclusiones obtenidas en este proceso de evaluación.

5.1 Proceso de evaluación

Este apartado corresponde a la primera parte de la evaluación del sistema. Este se compone a su vez de los sub-apartados: en el primero se escoge y diseña el método a evaluar, y el segundo se ejecuta dicho método.

Plan de pruebas

En este apartado se expone el plan de pruebas utilizado para la evaluación de este sistema. Lo primero que se va a realizar son los diferentes casos de prueba que permitan verificar que el sistema cumple con los requisitos establecidos en la fase de análisis. Más adelante, analizaremos los resultados obtenidos al verificar los casos de prueba.
En este caso, los casos de prueba están relacionados con los casos de uso definidos en el apartado de Casos de Uso presentados anteriormente. Debido a que todos los requisitos están contenidos en los casos de uso, si generamos los casos de prueba en función de los casos de uso podemos verificar en los requisitos cumplen el sistema que se ha desarrollado.

Definiremos los casos de prueba según esta plantilla:

CP-XX	
Nombre	
Descripción	
Caso de uso	
Precondición	

Secuencia	
Verificación	

Figura 34: Plantilla de Casos de Prueba

Casos de prueba

En este apartado, se va a proceder a presentar los casos de prueba siguiendo la plantilla del apartado anterior. Como ya mencionamos anteriormente, estos casos de pruebas tan basados en los casos de uso definidos en la fase de análisis del sistema. Estos escenarios se van a evaluar en el sistema y se va a verificar el resultado de los mismos.

A continuación se presentan los casos de prueba necesarios para evaluar la aplicación:

CP-01	
Nombre	Seleccionar conjunto de datos
Descripción	El usuario desea seleccionar el método de análisis
Caso de uso	CU-01
Precondición	El usuario debe estar logueado en la aplicación
Secuencia	1. El usuario despliega el menú del apartado Data Sets 2. El usuario selecciona el conjunto de datos 3. El usuario pulsa el botón apply Data Set
Verificación	Se muestran los atributos del conjunto de datos en campo Selection of Attributes

Figura 35: Caso de prueba: CP-01

CP-02	
Nombre	Seleccionar metodo
Descripción	El usuario desea seleccionar el conjunto de datos
Caso de uso	CU-02
Precondición	El usuario debe estar logueado en la aplicación
Secuencia	1. El usuario despliega el menú del apartado Method 2. El usuario selecciona el conjunto de datos 3. El usuario pulsa el botón apply
Verificación	Se muestran las opciones referentes al método seleccionado

Figura 36: Caso de prueba: CP-02

CP-03	
Nombre	Seleccionar atributos
Descripción	El usuario desea seleccionar los atributos del conjunto que desea evaluar
Caso de uso	CU-03
Precondición	El usuario debe estar logueado en la aplicación
Secuencia	1. El usuario selecciona los atributos deseados en el apartado Selection of Attributes 3. El usuario pulsa el botón run
Verificación	Se muestran los resultados referentes a los atributos que se han seleccionado

Figura 37: Caso de prueba: CP-03

CP-04	
Nombre	Identificación de usuarios
Descripción	El usuario desea acceder a la aplicación
Caso de uso	CU-04
Precondición	Ninguna
Secuencia	1. El usuario introduce sus credenciales en la pantalla de logeo 3. El usuario pulsa el botón Sign in
Verificación	Se muestra la pantalla principal

Figura 38: Caso de prueba: CP-04

CP-05	
Nombre	Selección de metodo de Test
Descripción	El usuario desea seleccionar el metodo de test del análisis
Caso de uso	CU-05
Precondición	El usuario debe estar logueado en la aplicación
Secuencia	1. El usuario despliega el menú de Test options 2. El usuario selecciona el test 3. El usuario pulsa el botón apply
Verificación	Se muestran los resultados referentes al método de test que se ha seleccionado

Figura 39: Caso de prueba: CP-05

CP-06	
Nombre	Selección atributo de clase
Descripción	El usuario desea seleccionar atributo que se tomará como clase
Caso de uso	CU-06
Precondición	El usuario debe estar logueado en la aplicación
Secuencia	1. El usuario despliega el menú de Class attribute 2. El usuario selecciona el atributo de clase 3. El usuario pulsa el botón apply
Verificación	Se muestran los resultados referentes al metodo de clasificación tomando como clase el atributo seleccionado

Figura 40: Caso de prueba: CP-06

CP-07	
Nombre	Selección número de agrupaciones o clusters
Descripción	El usuario desea seleccionar el número de agrupaciones o clusters del modelo
Caso de uso	CU-07
Precondición	El usuario debe estar logueado en la aplicación
Secuencia	1. El usuario despliega el menú de Number of clusters 2. El usuario selecciona el número deseado 3. El usuario pulsa el botón change clusters
Verificación	Se muestran los resultados referentes al método de agrupamiento tomando con el número de grupos seleccionados

Figura 41: Caso de prueba: CP-07

CP-08	
Nombre	Mostrar resultados
Descripción	El usuario recibirá los resultados del modelo generado
Caso de uso	CU-08
Precondición	El usuario debe haber generado un modelo
Secuencia	1. El usuario genera un modelo 2. El usuario pulsa el botón run 3. El usuario recibe los resultados en el campo Results
Verificación	Se muestran los resultados referentes al método y atributos seleccionados

Figura 42: Caso de prueba: CP-08

CP-09	
Nombre	Mostrar árbol
Descripción	El usuario recibirá el árbol del modelo generado
Caso de uso	CU-09
Precondición	El usuario debe haber generado un modelo
Secuencia	1. El usuario genera un modelo 2. El usuario pulsa el botón run 3. El usuario recibe el árbol en el campo Results
Verificación	Se muestran en los resultados el árbol de clasificación

Figura 43: Caso de prueba: CP-09

CP-10	
Nombre	Mostrar grupos de pertenencia
Descripción	El usuario recibirá los grupos a los que pertenece cada instancia del conjunto de datos
Caso de uso	CU-10
Precondición	El usuario debe haber generado un modelo
Secuencia	1. El usuario genera un modelo 2. El usuario pulsa el botón run 3. El usuario recibe la información en el campo Results
Verificación	Se muestran en los resultados cada una de las tuplas con los atributos seleccionados y al lado aparece el cluster de permanencia

Figura 44: Caso de prueba: CP-10

CP-11	
Nombre	Registrarse
Descripción	El usuario podrá registrarse en la aplicación
Caso de uso	CU-11
Precondición	Ninguna
Secuencia	1. El usuario pulsa el botón Sign up en la pantalla de logueo 2. El usuario introduce su email 3. El usuario introduce la contraseña en el campo password 4. El usuario introduce nuevamente la contraseña en el campo password confirmation 5. El usuario pulsa el botón Sign up
Verificación	Se muestra la pantalla principal

Figura 45: Caso de prueba: CP-11

5.2 Análisis de resultados

En este apartado se analizan los resultados obtenidos en las pruebas descritas el apartado anterior. En el análisis de los resultados se va formalizar por medio de una tabla en la que se recojan los resultados obtenidos en el proceso de evaluación. Para cada prueba asociada se especificará la fecha en la que se realiza y el resultado de la prueba. En caso de no tener un resultado satisfactorio, será necesario añadir una descripción especificando los resultados que se obtienen.
A continuación se muestran los resultados obtenidos en el proceso de evaluación de los casos de prueba:

Caso de prueba	Fecha	Resultado	Descripción
CP-01	15/08/2014	Satisfactorio	
CP-02	15/08/2014	Satisfactorio	
CP-03	15/08/2014	Satisfactorio	
CP-04	15/08/2014	Satisfactorio	
CP-05	15/08/2014	Satisfactorio	
CP-06	16/08/2014	Satisfactorio	
CP-07	16/08/2014	Satisfactorio	
CP-08	17/08/2014	Satisfactorio	
CP-09	18/08/2014	Satisfactorio	
CP-10	19/08/2014	Satisfactorio	
CP-11	19/08/2014	Satisfactorio	

Figura 46: Evaluación de los Casos de Prueba

Como se puede observar en los resultados obtenidos, todos los casos de uso han sido verificados en el proceso de evaluación, por lo que podemos dar por verificada la funcionalidad del sistema.

6 Conclusions

This section summarizes the main contributions of the project, lists the problems encountered along their development, discuss possible future work based on the current development and personal reviews of the work are given.

6.1 Contributions

The main contribution of this project is the development of a client-server application that allows to perform predictive analysis with the data of the e-commerce platform Magento, in order to support business decisions. As a result, this project extends the statistical report capacity of the management section in the Magento platform, allowing the user to obtain customized reports by means of the analysis of the data of its customers and products. Note that this application is a proof of concept for what brand guidelines to follow in order to extend the functionality of the system with little effort.

6.2 Future works

In this section are enumerated some future works that could be derived from this Project.

The first step should be tracked to develop the necessary models to generate basic data structures for the main components of an e-commerce application i.e. mainly would add a sales model and a model for catalogues in order to complete the analysis capability of the developed system. This analysis could be made on a set of data and compare with similar structures in other models to expand wealth of reports and better support business decisions.

The second step would be include more data set attributes. If we add a greater number of attributes it would expand the quality of the retrieved reports.

Another interesting step would be to expand the predictive analysis techniques implemented in the system. This enables the system to carry out **prediction** and not be very expensive because it is already developed the principal pattern. Because this application is developed based on the Model-View-Controller architecture it would not be extremely difficult.

The next thing to consider when expanding system functionality would include all machine learning models integrated into the tool, to get higher completeness and comparative capacity with the systems of the same type. This is object of our analysis because the same report is not obtained by acting with different classifiers and these nuances can determine better reporting.

On the other hand, another measure to consider is that directed to include processes of asynchronous communication between the front-end and back-end, thus avoiding having to make a submit whenever any data is changed the dropdown menus. This would provide a better view of the system and significantly improve the user interaction experience.

Another measure would be to develop the integration of this application as a module of the e-commerce platform Magento getting add this functionality directly to the platform and not use an external system acting on Magento data. This would provide better speed and access to information and wouldn't need to use the identification process because it would be enough with administrative access to the own platform.

6.3 Problems encountered

Referring to the data model, the main problems have arisen when trying to understand the data model used by Magento. The EAV model presents a structure which I had never worked, this implied much analysis time. Being a model, as explained in section 2.3.2 Modelo de datos EAV, in which the attributes are not linked to entities, to track each of the data related to the attributes stored in specific tables has meant spend much time in the phases of design and implementation.

Reference at development of the web application, the main problems arise mainly because of the experience in technology used. Ruby has its own unique coding language and this is a disadvantage because there is a significant time taken to learn it. Another of the principal problems of developing an application in the Ruby programming language is that it has little support community online and there are few books published. Ruby is a relatively new technology This technology does not have communities up to the most popular languages like PHP and C #. According MSDN [MSDN], a blog of Microsoft Corporation, from 2006, about 400 books on C # were on the market, while only related to the amount of Ruby 50.

The main problem encountered when developing the application is linked to the problems outlined above. Because of the lack of experience in the programming language, poor documentation and complexity of the EAV data model, certainly the most difficult task has been to adapt the Magento data model of the application developed. They had to generate several auxiliary models for mapping the data in Magento model and work with that data following the coding paradigm Ruby on Rails.
Another problem that arose during the development was the communication between Views and

Controllers due to two fundamental problems: the MVC model is a model that we had not worked before, so I was not totally sure about this architecture; and the way to treat the event through a naming convention and route system that works with Ruby on Rails. If we add the missing documentation, it quite complicated to implement system interaction.

6.4 Personal opinions

The realization of this bachelor project has been a great challenge. In particular, as discussed in the previous section, there have been many problems in the development of the application result of the lack of experience and documentation development platform. However, the planning of the project has managed to accomplish in stipulated time and have acquired new skills and capabilities through its realization. Concluding this section, I would like to highlight some of the skills that have been obtained, along Degree in Computer Engineering and during the development of this Bachelor Project.

Throughout the degree, I have gained many skills that I have put into practice and have been very helpful in making this final Bachelor Project. One of these skills is the ability to generate a proper data model. This design has been generated through the knowledge acquired in the course "Database Design" course and has vital knowledge for the successful development of this project. Another subject that has proved very useful for the development of project documentation, has been "Project Management and Software Development", in which I have worked with each and every one of the elements present in this thesis; from a correct definition of "State of Art" to the generation of a test plan, to each of the relevant phases of a software project.

Through this final project, I have also obtained numerous capacities. One of the main skills has been to work and adapt data between two different models. I never had to develop a middleware component in the application to map data between these two models.
Another of the knowledge obtained during the course of this project, has been working on a completely new programming language for me, Ruby on Rails. This has given me the ability to work beneath the MVC architecture and modular programming paradigm following a very specific name convention, which has given me a greater ability to structure and development organization.

7 Bibliografía

Referencias utilizadas para la realización del trabajo.

[WKA] "Weka 3". Machine Learning Group. University of Waikato. 2013. Disponible [Internet]: http://www.cs.waikato.ac.nz/ml/weka/

[APR] "Analisis PredictivoFundación BigData. 2014. Disponible [Internet]: http://fundacionbigdata.org/analisis-predictivo/

[BIN] Chen, H., Chiang, R. H., & Storey, V. C. (2012). Business Intelligence and Analytics: From Big Data to Big Impact. MIS quarterly, 36(4), 1165-1188. Disponible [Internet]: http://hmchen.shidler.hawaii.edu/Chen_big_data_MISQ_2012.pdf

[MLR] Chan, J. C. W., Chan, K. P., & Yeh, A. G. O. (2001). Detecting the nature of change in an urban environment: A comparison of machine learning algorithms. Photogrammetric Engineering and Remote Sensing, 67(2), 213-225. Disponible [Internet]: http://hub.hku.hk/handle/10722/118210

[RMN] "RapidMiner" A.I. Department Dortmund University.2001. Disponible [Internet]: http://rapidminer.com/

[MGN] "Magento" Varient. 2008. Disponible [Internet]: http://magento.com/

[PRS] "Prestashop" Prestashiop Group.2008. Disponible [Internet]: http://www.prestashop.com/es/

[EAV] Ganslandt, T., Mueller, M., Krieglstein, C. F., Senninger, N., & Prokosch, H. U. (1999). A flexible repository for clinical trial data based on an entity-attribute-value model. In Proceedings of the AMIA Symposium (p. 1064). American Medical Informatics Association. Disponible [Internet]: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2232837/

[AFX] "Adobe Flex". Adobe Systems Incorporated. 2014. Disponible [Internet]: http://www.adobe.com/es/products/flex.html

[ROR] "Ruby On Rails" David Heinemeier Hansson. 2005. Disponible [Internet]:
http://rubyonrails.org/

[BTC] "Bases y tipos de cotización contingencias comunes" Ministerio de Empleo y Seguridad Social. 2014. Disponible [Internet]:
http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Basesytiposdecotiza36537/index.htm

[ESP] "Modelo en Espiral" Ing. Jymmy Guevara. 2012. Disponible [Internet]:
https://sites.google.com/site/adai6jfm/modelo-en-espiral

[VOL] Volere, "Plantilla de Especificación de Requisitos".2006. Disponible [Internet]:
http://www.volere.co.uk/pdf%20files/template_es.pdf

[MIG] "Migrations Rails" Guide Ruby On Rails.2008. Disponible [Internet]:
http://guides.rubyonrails.org/migrations.html

[MVC] Deacon, J. (2009). "Model-view-controller (mvc) architecture" .2006. Disponible [Internet]:
http://www.jdl.co.uk/briefings/MVC.pdf

[ACR] "Active Record Basics Rails" Guide Ruby On Rails.2008. Disponible [Internet]:
http://guides.rubyonrails.org/active_record_basics.html

[MSDN] "Microsoft Development Network". Disponible [Internet]:
http://msdn.microsoft.com/es-ES/

Anexo I. Control de versiones

Versión	Fecha de finalización	Descripción
1.0	25/08/2014	Redacción del apartado 2 "El estado de la cuestión"
2.0	08/09/2014	Redacción de los apartados 1 y 3
3.0	14/09/2014	Redacción del apartado 4 "Solución"
4.0	18/09/2014	Redacción del apartado 5 "Evaluación"
5.0	20/09/2014	Redacción de los apartados 6 y Anexos. Revisión de los apartados anteriores
6.0	22/09/2014	Revisión de los apartados 6 y Anexos. Corrección de los apartados 1 - 5. Maquetación Final
7.0	23/09/2014	Revisión y correcciones de los apartados en inglés. Maquetación Final

Tabla 19: Control de versiones

Anexo II. Seguimiento de proyecto fin de carrera

En este apartado se detalla el seguimiento del presente trabajo fin de grado. En primer lugar, se establece la forma de seguimiento que se va a realizar para, a continuación, especificar la planificación inicial del proyecto, especificando las tareas realizadas y sus plazos. El apartado finaliza con unas líneas en las que se indica la fidelidad respecto de la planificación inicial.

Forma de seguimiento

La forma de seguimiento del trabajo fin de grado ha consistido en reuniones mensuales con el tutor del proyecto. Las reuniones se han mantenido durante los meses comprendidos entre Enero y Junio y se han retomado en Septiembre. En estas reuniones se exponían los avances y se establecían los hitos a cumplir para la siguiente reunión. Durante Julio y Agosto el seguimiento se limitó a la presentación de los avances por medio de plataformas electrónicas y resolución de dudas por medio de correo electrónico. Por cuestiones técnicas y personales, estas reuniones se han podido adelantar o retrasar, aunque nunca se han retrasado más de una semana.

Planificación inicial

La planificación del trabajo fin de grado se ha formalizado mediante un diagrama de Gantt en el que se especifican las tareas realizadas y las fechas de inicio y de fin. La planificación se ha realizado teniendo en cuenta la ejecución completa del proyecto, pasando por todas sus fases. A continuación se muestra el diagrama en cuestión:

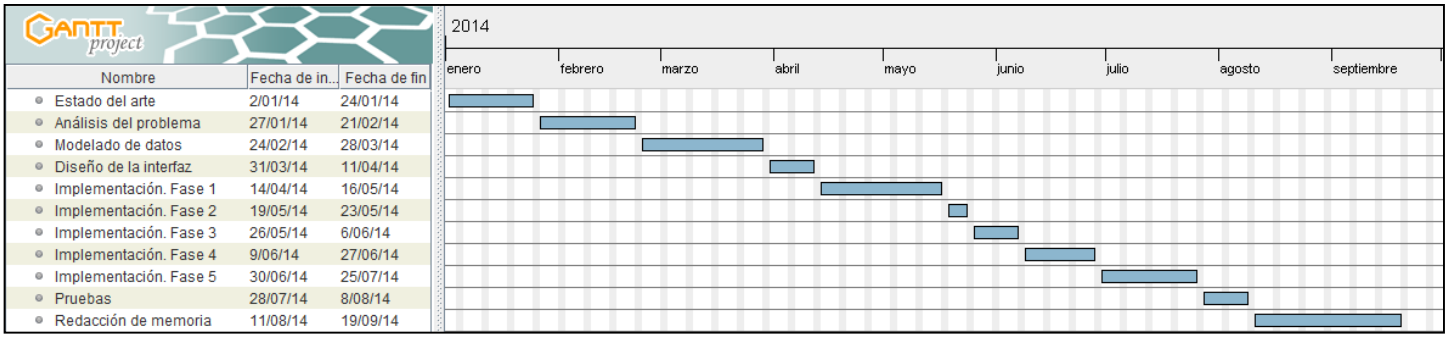


Figura 47: Diagrama de Gantt

A continuación se especifican las tareas que componen el proyecto:

- **Estado del arte:** se define el problema a resolver por el proyecto y se establece un contexto en el que se sitúa. En última instancia, se realiza una comparativa entre las tecnologías y herramientas analizadas para escoger las más adecuadas.

- **Análisis del problema:** se describen las principales características del problema a resolver por medio de la identificación de **requisitos**.

- **Modelado de datos:** se estudiará el modelo de datos de la plataforma Magento para generar un modelo de datos adaptable a la plataforma que se va a implementar.

- **Diseño de la interfaz:** se define la interfaz del sistema, especificando todos los componentes (transiciones, interfaces, componentes) que se desarrollan en el entorno.

- **Implementación. Fase 1:** esta fase hace referencia a la implementación del modelo de datos. **S**e desarrollan los componentes referentes al modelo de datos estudiados en la fase de Modelado de datos, es decir, los conjuntos de datos que se recogen de la plataforma Magento.

- **Implementación. Fase 2:** esta fase hace referencia a la implementación los componentes visuales. Se desarrollan los componentes referentes a la interfaz de usuario.

- **Implementación Fase 3:** esta fase hace referencia a la implementación de la funcionalidad de la aplicación. Se realiza la integración de los componentes visuales con el modelo de datos.

- **Implementación Fase 4:** esta fase hace referencia a la integración de la herramienta de generación de modelos con la Fase 3, para cerrar una versión funcional de la aplicación.

- **Implementación. Fase 5:** esta fase hace referencia a la implementación modificadores en la aplicación para realizar el preprocesado de los datos y aportar opciones a la hora de generar los modelos.

- **Pruebas:** esta fase corresponde con las pruebas que se realizan sobre la aplicación siguiendo el plan de pruebas especificado.

- **Redacción de memoria:** en esta fase se elabora el presente documento, correspondiente a la documentación del proyecto completo.

Planificación final

El presente trabajo ha cumplido la planificación inicial, habiendo finalizado todas las etapas en el tiempo estimado y habiendo sido entregado en fecha, cumpliendo los objetivos propuestos.

Anexo III. Especificación de Requisitos

En este punto se analizarán y especificarán los requisitos de la forma descrita en el apartado de Análisis según la adaptación de la plantilla requisitos de Volere [VOL].

Requisitos funcionales

RF – 01			
Nombre	Selección del conjunto de datos		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	permitirá seleccionar el conjunto de datos		

Tabla 20: Requisito funcional RF-01

RF – 02			
Nombre	Selección del método		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	Se permitirá seleccionar el método de aplicación		

Tabla 21: Requisito funcional RF-02

RF – 03			
Nombre	Selección de atributos del conjunto de datos		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	permitirá seleccionar los atributos del conjunto de datos con los que desea operar		

Tabla 22: Requisito funcional RF-03

RF – 04	
Nombre	Identificación de usuarios
Fuente	Cliente

Prioridad	Media	Claridad	Alta
Descripción	permitirá la identificación de usuarios		

Tabla 23: Requisito funcional RF-04

RF – 05			
Nombre	Selección de método de test		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	Permitirá seleccionar el método de test cuando utilice un método de clasificación.		

Tabla 24: Requisito funcional RF-05

RF – 06			
Nombre	Selección atributo de clase		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	permitirá la selección del atributo de clase cuando utilice el método de clasificación		

Tabla 25: Requisito funcional RF-06

RF – 07			
Nombre	Selección número de agrupaciones		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	permitirá la selección el número de agrupaciones cuando se utilice el método agrupamiento		

Tabla 26: Requisito funcional RF-07

RF – 08			
Nombre	Mostrar resultados		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	Devolverá los resultados obtenidos		

Tabla 27: Requisito funcional RF-08

RF – 09			
Nombre	Mostrar Árbol		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	Mostrará el árbol cuando se ejecute una tarea de clasificación		

Tabla 28: Requisito funcional RF-09

RF – 10			
Nombre	Muestra de grupos		
Fuente	Cliente		
Prioridad	Alta	Claridad	Media
Descripción	Mostrará los grupos a los que pertenece cada instancia del conjunto de datos		

Tabla 29: Requisito funcional RF-10

RF – 11			
Nombre	Registro		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	Permitirá registrarse en la aplicación		

Tabla 30: Requisito funcional RF-11

Requisitos no funcionales

RNF – 01			
Nombre	Idioma interfaz		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	Mostrará la interfaz en ingles		

Tabla 31: Requisito no funcional RFN-01

RNF – 02	
Nombre	Compatibilidad

Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	Deberá ser desplegable en cualquier navegador		

Tabla 32: Requisito no funcional RNF-02

RNF – 03			
Nombre	Sencillez en el interfaz		
Fuente	Cliente		
Prioridad	Alta	Claridad	Media
Descripción	Deberá mostrar una interfaz sencilla		

Tabla 33: Requisito no funcional RNF-03

RNF – 04			
Nombre	Interfaz en columnas		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	La interfaz contará con cuatro columnas: Data Set, Method Selection of Attributes y Results		

Tabla 34: Requisito no funcional RNF-04

RNF – 05			
Nombre	Menú desplegable		
Fuente	Cliente		
Prioridad	Baja	Claridad	Alta
Descripción	La selección de conjuntos se realizará por medio de un menú desplegable		

Tabla 35: Requisito no funcional R--F05

RNF – 06			
Nombre	Visualización		
Fuente	Cliente		
Prioridad	Media	Claridad	Media
Descripción	Será visualizado un en un monitor de mínimo 15 pulgadas		

Tabla 36: Requisito no funcional RNF-06

RNF – 07			
Nombre	Selección única del conjunto de datos		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	permitirá seleccionar un solo conjunto de datos con cada ejecución del modelo		

Tabla 37: Requisito no funcional RNF-06

RNF – 08			
Nombre	Selección única del método		
Fuente	Cliente		
Prioridad	media	Claridad	Alta
Descripción	Solo permitirá seleccionar un método por cada ejecución		

Tabla 38: Requisito no funcional RNF-08

RNF – 09			
Nombre	Selección múltiple de atributos del conjunto de datos		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	permitirá seleccionar los atributos del conjunto de datos de forma múltiple		

Tabla 39: Requisito no funcional RNF-09

RNF – 10			
Nombre	Selección entre métodos de test		
Fuente	Cliente		
Prioridad	Media	Claridad	Alta
Descripción	permitirá la selección el método de test entre cross validation y test set		

Figura 48: Requisito no funcional RNF-10

RNF – 11			
Nombre	Selección entre atributos de clase		
Fuente	Cliente		
Prioridad	Media	Claridad	Media
Descripción	permitirá la selección del atributo de clase entre los atributos del conjunto de datos seleccionado		

Tabla 40: Requisito funcional RNF-11

RNF – 12			
Nombre	Limites número de agrupaciones		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	permitirá la selección un número de agrupaciones contenido entre 2 y 5		

Tabla 41: Requisito funcional RNF-12

RNF – 13			
Nombre	Coordinación		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	Existirá coordinación entre los modelos de datos y los datos contenidos en la base de datos de Magento		

Tabla 42: Requisito funcional RNF-13

RNF – 14			
Nombre	Conjuntos de datos		
Fuente	Cliente		
Prioridad	Alta	Claridad	Media
Descripción	Contará con 2 conjuntos de datos: Products y Customers		

Tabla 43: Requisito funcional RNF-14

RNF – 15	
Nombre	Coordinación credenciales

Fuente	Cliente		
Prioridad	Alta	Claridad	Media
Descripción	Existirá coordinación entre las credenciales introducidas en la aplicación y los usuarios almacenados en la base de datos		

Tabla 44: Requisito funcional RNF-15

RNF – 16			
Nombre	Conjunto Products		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	El conjunto de datos de Products contará con los siguientes atributos: product_id, name, code, weirht, country, quantity, price y category.		

Tabla 45: Requisito funcional RNF-16

RNF – 17			
Nombre	Conjunto Customers		
Fuente	Cliente		
Prioridad	Alta	Claridad	Alta
Descripción	El conjunto de datos Customers contará con las siguientes variables: customer_id, name, surname, birthdate, pone, country, province, city, postalcode y gender		

Tabla 46: Requisito funcional RNF-17

Anexo IV. Especificación de Casos de Uso

Para la presentación de los casos de uso utilizaremos esta plantilla:

CU-XX	
Nombre	
Actores	
Descripción	
Precondiciones	
Curso básico	

Tabla 47: Plantilla Casos de Uso

A continuación se presentan los Casos de Uso extraídos de los requisitos:

CU-01	
Nombre	Seleccionar conjunto de datos
Actores	Usuario registrado
Descripción	El usuario desea seleccionar el conjunto de datos
Precondiciones	El usuario debe tener una cuenta en la aplicación
Curso básico	1. El usuario despliega el menú del apartado Data Sets 2. El usuario selecciona el conjunto de datos 3. El usuario pulsa el botón apply Data Set

Tabla 48: Caso de Uso CU-01

CU-02	
Nombre	Seleccionar método
Actores	Usuario registrado
Descripción	El usuario desea seleccionar el método de análisis
Precondiciones	El usuario debe tener una cuenta en la aplicación
Curso básico	1. El usuario despliega el menú del apartado Method 2. El usuario selecciona el conjunto de datos 3. El usuario pulsa el botón apply

Tabla 49: Caso de Uso CU-02

CU-03	
Nombre	Seleccionar atributos
Actores	Usuario registrado
Descripción	El usuario desea seleccionar los atributos del conjunto que desea evaluar
Precondiciones	El usuario debe tener una cuenta en la aplicación
Curso básico	1. El usuario selecciona los atributos deseados en el apartado Selection of Attributes 2. El usuario pulsa el botón run

Tabla 50: Caso de Uso CU-03

CU-04	
Nombre	Identificación de usuarios
Actores	Usuario
Descripción	El usuario desea acceder a la aplicación
Precondiciones	El usuario debe tener una cuenta en la aplicación
Curso básico	1. El usuario introduce el email y la contraseña en la pantalla de logeo 2. El usuario pulsa el botón Sign in

Tabla 51: Caso de Uso CU-04

CU-05	
Nombre	Selección de método de Test
Actores	Usuario registrado
Descripción	El usuario desea seleccionar el método de test del análisis
Precondiciones	El usuario debe tener una cuenta en la aplicación
Curso básico	1. El usuario despliega el menú de Test options 2. El usuario selecciona el test 3. El usuario pulsa el botón apply

Tabla 52: Caso de Uso CU-05

CU-06	
Nombre	Selección atributo de clase
Actores	Usuario registrado
Descripción	El usuario desea seleccionar atributo que se tomará como clase.
Precondiciones	El usuario debe tener una cuenta en la aplicación

Curso básico	1. El usuario despliega el menú de Class attribute 2. El usuario selecciona el atributo de clase 3. El usuario pulsa el botón apply

Tabla 53: Caso de Uso CU-06

CU-07	
Nombre	Selección número de agrupaciones o clusters
Actores	Usuario registrado
Descripción	El usuario desea seleccionar el número de agrupaciones o clusters del modelo.
Precondiciones	El usuario debe tener una cuenta en la aplicación
Curso básico	1. El usuario despliega el menú de Number of clusters 2. El usuario selecciona el número deseado 3. El usuario pulsa el botón change clusters

Tabla 54: Caso de Uso CU-07

CU-08	
Nombre	Mostrar resultados
Actores	Usuario registrado
Descripción	El usuario recibirá los resultados del modelo generado
Precondiciones	El usuario debe de haber creado un modelo
Curso básico	1. El usuario genera un modelo 2. El usuario pulsa el botón run 3. El usuario recibe los resultados en el campo Results

Tabla 55: Caso de Uso CU-08

CU-09	
Nombre	Mostrar árbol
Actores	Usuario registrado
Descripción	El usuario recibirá el árbol del modelo generado
Precondiciones	El usuario debe de haber creado un modelo
Curso básico	1. El usuario genera un modelo 2. El usuario pulsa el botón run 3. El usuario recibe el árbol en el campo Results

Tabla 56: Caso de Uso CU-09

CU-10	
Nombre	Mostrar grupos de pertenencia
Actores	Usuario registrado
Descripción	El usuario recibirá los grupos a los que pertenece cada instancia del conjunto de datos
Precondiciones	El usuario debe de haber creado un modelo
Curso básico	1. El usuario genera un modelo 2. El usuario pulsa el botón run 3. El usuario recibe la información en el campo Results

Tabla 57: Caso de Uso CU-10

CU-11	
Nombre	Registrarse
Actores	Usuario
Descripción	El usuario podrá registrarse en la aplicación
Precondiciones	Ninguna
Curso básico	1. El usuario pulsa el botón Sign up en la pantalla de logueo 2. El usuario introduce su email 3. El usuario introduce la contraseña en el campo password 4. El usuario introduce nuevamente la contraseña en el campo password confirmation 5. El usuario pulsa el botón Sign up

Tabla 58: Caso de Uso CU-11